

On Multi-source Networks: Enumeration, Rate Region Computation, and Hierarchy

Congduan Li, *Member, IEEE*, Steven Weber, *Senior Member, IEEE*, and
John MacLaren Walsh, *Member, IEEE*.

Abstract—Recent algorithmic developments have enabled computers to automatically determine and prove the capacity regions of small hypergraph networks under network coding. A structural theory relating network coding problems of different sizes is developed to make best use of this newfound computational capability. A formal notion of network minimality is developed which removes components of a network coding problem that are inessential to its core complexity. Equivalence between different network coding problems under relabeling is formalized via group actions, an algorithm which can directly list single representatives from each equivalence class of minimal networks up to a prescribed network size is presented. This algorithm, together with rate region software, is leveraged to create a database containing the rate regions for all minimal network coding problems with five or fewer sources and edges, a collection of 744119 equivalence classes representing more than 9 million networks. In order to best learn from this database, and to leverage it to infer rate regions and their characteristics of networks at scale, a hierarchy between different network coding problems is created with a new theory of combinations and embedding operators.

I. INTRODUCTION

Determining the capacity regions of networks under network coding form a highly important class of problems according to perspectives both theoretical and applied. Indeed, these problems are of fundamental importance in multi-terminal information theory, not only because they are the simplest – with independent sources, perfect channels, and lossless reconstruction – variants of general multiterminal problems, but also because solving all of these problems is equivalent to determining all of the fundamental laws of information theory [4]–[7]. From a more practical viewpoint, optimized designs for many important modern engineering problems, including efficient information transfer over networks [8], [9], the design of efficient distributed information storage systems [10], [11], and the design of streaming media systems [12]–[14], have been shown to involve determining the rate region of an abstracted network under network coding. Yan *et al.*'s celebrated paper [15] has provided an exact representation of these rate regions of networks under network coding. Their essential result is that the rate region of a network can be expressed as the intersection of the region of entropic vectors

[16], [17] with a series of linear (in)equality constraints created by the network's topology and the sink-source requirements, followed by a projection of the result onto the entropies of the sources and edge variables. However, this is only an implicit description of the rate region, because the region of entropic vectors $\bar{\Gamma}_N^*$ is still unknown for $N \geq 4$.

Nevertheless, recent algorithmic developments have exploited this implicit characterization to enable computers to automatically determine and prove the capacity regions of small hypergraph networks under network coding [18]–[22]. Thanks to these algorithms and their implementations in packages like the *information theoretic converse prover (ITCP)* [23] and the *information theoretic achievability prover (ITAP)* [24], it is now possible to very rapidly calculate the rate region, its proof, and the class of capacity achieving codes, for small networks, each of which would previously have taken a trained information theorist hours or longer to derive. The availability of these methods does not diminish the role of traditional theory proved by hand, rather, it shifts the questions driving theory development in new directions.

This article sets about developing a structural theory organizing and relating network coding problems of different sizes, whose aim is to make most efficient use of this newfound computational capability to determine capacity regions and their properties of networks of arbitrary size and scale. This structural theory consists of three components: minimality, symmetry, and hierarchy. In §III, a formal notion of network coding problem minimality is presented, which precisely describes how to remove from a network coding problem components which are redundant or irrelevant. For each such reduction, a theorem shows how the larger, non-minimal network, has a capacity region which can be directly inferred from the smaller one. The next component of the theory, symmetry, discussed in §IV, observes that in order to precisely formulate a network coding problem, labels must be applied to sources and edges, however the underlying problem is completely insensitive to the selection of these labels. Properly formulating the associated symmetry group through the language of group actions, together with a concise problem description aided by the notion of a minimal network, enable an algorithm for directly listing only the canonical and minimal representative of each large class of equivalent networks to be derived. This listing algorithm is used, together with the automated rate region calculation algorithms, in §V to generate a massive database containing the capacity regions of all hypergraph networks with the sum of sources and edges less than or equal to five. It is shown that for these small

Support under National Science Foundation awards CCF-1016588 and 1421828 is gratefully acknowledged.

C. Li, S. Weber and J. W. Walsh are with the Department of Electrical and Computer Engineering, Drexel University, Philadelphia, PA USA (email: congduan.li@drexel.edu, sweber@coe.drexel.edu, and jwalsh@coe.drexel.edu). Preliminary results were presented at Allerton 2014 [1], NetCod 2015 [2], and ITW 2015 [3].

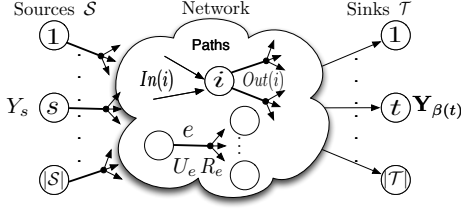


Figure 1: A general network model A .

problems, linear codes suffice to exhaust the rate region and the Shannon outer bound is tight.

Finally, a hierarchical theory capable of inferring rate regions and properties from larger networks from smaller ones contained within them is presented. First, §VI, embedding operations, which recognize a small network coding problem embedded in a larger one, are defined and utilized, together with key theorems tracking rate regions through embeddings, to create the notion of forbidden network minors. These enable properties such as lack of sufficiency of a class of linear codes, or lack of tightness of the Shannon outer bound, to be determined for arbitrarily large networks by testing for inclusion of certain small problems within them. Next, desiring to construct the efficient codes of rate regions for large network coding problems by viewing them as being constructed from those of smaller networks, in §VII a series of combination operators are presented. §VIII then shows that together, the combinations and embedding operators enable solutions for large networks to be constructed through simple operations using rate regions of small networks as building blocks. The power of this hierarchy is demonstrated both by organizing the database, and showing how it, coupled with the database, can determine the capacity regions and their properties for massive numbers of networks of arbitrary scale.

II. BACKGROUND: NETWORK CODING PROBLEM MODEL, CAPACITY REGION, AND BOUNDS

The class of problems under study in this paper are the rate regions of multi-source multi-sink network coding problems with hyperedges, which we hereafter refer to as the hyperedge MSNC problems. A network coding problem in this class, denoted by the symbol A , includes a directed acyclic hypergraph $(\mathcal{V}, \mathcal{E})$ [25] as in Fig. 1, consisting of a set of nodes \mathcal{V} and a set \mathcal{E} of directed hyperedges in the form of ordered pairs $e = (v, \mathcal{A})$ with $v \in \mathcal{V}$ and $\mathcal{A} \subseteq \mathcal{V} \setminus v$. The nodes \mathcal{V} in the graph are partitioned into the set of source nodes \mathcal{S} , intermediate nodes \mathcal{G} , and sink nodes \mathcal{T} , i.e., $\mathcal{V} = \mathcal{S} \cup \mathcal{G} \cup \mathcal{T}$. Each of the source nodes $s \in \mathcal{S}$ will have a single outgoing edge $(s, \mathcal{A}) \in \mathcal{E}$. The source nodes in \mathcal{S} have no incoming edges, the sink nodes \mathcal{T} have no outgoing edges, and the intermediate nodes \mathcal{G} have both incoming and outgoing edges.

The number of sources will be denoted by $|\mathcal{S}| = K$, and each source node $s \in \mathcal{S}$ will be associated with an independent random variable Y_s , $s \in \mathcal{S}$, with entropy $H(Y_s)$, and an associated independent and identically distributed (IID) temporal sequence of random values. For every source $s \in \mathcal{S}$, define $\text{Out}(s)$ to be its single outgoing edge, which is connected to a subset of intermediate nodes and sink nodes. An edge $e \in \mathcal{E}$ connects a source, or an intermediate node to a subset

of non-source nodes, i.e., $e = (i, \mathcal{F})$, where $i \in \mathcal{S} \cup \mathcal{G}$ and $\mathcal{F} \subseteq (\mathcal{G} \cup \mathcal{T} \setminus i)$. For an intermediate node $g \in \mathcal{G}$, we denote its incoming edges as $\text{In}(g)$ and outgoing edges as $\text{Out}(g)$.

For each edge $e = (i, \mathcal{F})$, the associated random variable $U_e = f_e(\text{In}(i))$ is a function of all the inputs of node i , obeying the edge capacity constraint $R_e \geq H(U_e)$. The tail (head) node of edge e is denoted as $\text{Ti}(e)$ ($\text{Hd}(e)$). For notational simplicity, the unique outgoing edge of each source node will be the source random variable, $U_e = Y_s$ if $\text{Ti}(e) = s$, denoting $\mathcal{E}_S = \{e \in \mathcal{E} | \text{Ti}(e) = s, s \in \mathcal{S}\}$ to be the variables associated with outgoing edges of sources, and $\mathcal{E}_U = \mathcal{E} \setminus \mathcal{E}_S$ to be the non-source edge random variables.

For each sink $t \in \mathcal{T}$, the collection of sources this sink will demand will be labeled by the non-empty set $\beta(t) \subseteq \mathcal{S}$. Collecting these definitions, a network can be represented as a tuple $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$, where $\beta = (\beta(t), t \in \mathcal{T})$. For convenience, networks with K sources and $L = |\mathcal{E}_U|$ edges are referred as (K, L) instances.

As our focus in the manuscript will be on rate regions of a very similar form to those in [15], [17], this network coding problem model is as close to the original one in [15], [17] as possible while concisely covering the multiple instances of applications in network coding in which the same message can be overheard by multiple parties. These applications include index coding, wireless network coding, independent distributed source coding, and distributed storage, which are all covered concisely with this model as detailed in [26], [27].

It is in principle possible to transform the constraint of the same message being overheard by multiple parties into a directed acyclic graph used in [15], [17] by replacing each hyper-edge $e = (i, \mathcal{F})$ in our model with a new vertex v' and $|\mathcal{F}| + 1$ edges, $\{(i, v')\} \cup \{(v', j) | j \in \mathcal{F}\}$. However, this transformation effectively creates a far more difficult and complicated rate region, as each of the $|\mathcal{F}|$ new edges $\{(v', j) | j \in \mathcal{F}\}$ must also have associated new random variables and new rate constraints in the model of [15], [17]. As the complexity of determining the rate region through the implicit characterization in [15], [17] is at least exponentially dependent on the number of random variables, such a transformation of a problem with, in truth, a concise hypergraph structure, into a larger more complicated ordinary directed graph one is ill-advised. It is for this reason we depart notationally from the commonly used model from [15], [17]. Aside from this minor difference, the notion of a network code, an achievable rate vector, and the capacity region can be defined [26], [27] in an analogous manner to [15], [17].

Defining $\mathcal{L}_i, i = 1, 3, 4', 5$ as network constraints representing source independence, coding by intermediate nodes, edge capacity constraints, and sink nodes decoding constraints respectively,

$$\begin{aligned} \mathcal{L}_1 &= \{[\mathbf{h}^T, \mathbf{r}^T]^T : h_{\mathbf{Y}_S} = \sum_{s \in \mathcal{S}} h_{Y_s}\} \\ \mathcal{L}_3 &= \{[\mathbf{h}^T, \mathbf{r}^T]^T : h_{\mathbf{U}_{\text{Out}(g)}} | (\mathbf{Y}_{\mathcal{S} \cap \text{In}(g)} \cup \mathbf{U}_{\mathcal{E}_U \cap \text{In}(g)}) = 0, g \in \mathcal{G}\} \\ \mathcal{L}_{4'} &= \{[\mathbf{h}^T, \mathbf{r}^T]^T : R_e \geq h_{U_e}, \forall e \in \mathcal{E}_U\} \\ \mathcal{L}_5 &= \{[\mathbf{h}^T, \mathbf{r}^T]^T : h_{\mathbf{Y}_{\beta(t)} | \mathbf{U}_{\text{In}(t)}} = 0, \forall t \in \mathcal{T}\}, \end{aligned}$$

and denoting $\mathcal{L}_{13} = \mathcal{L}_1 \cap \mathcal{L}_3$, $\mathcal{L}_{4'5} = \mathcal{L}_{4'} \cap \mathcal{L}_5$ and $\mathcal{L}_A =$

$\mathcal{L}_1 \cap \mathcal{L}_3 \cap \mathcal{L}_{4'} \cap \mathcal{L}_5$, the implicit characterization from [15], [17] is translated in [26], [27] to the following one.

Theorem 1: *The rate region of a network A is expressible as*

$$\mathcal{R}_c(\mathbf{A}) = \text{Proj}_{\mathbf{r}, \omega}(\overline{\text{con}(\Gamma_N^* \cap \mathcal{L}_{13})} \cap \mathcal{L}_{4'5}), \quad (1)$$

where $\text{con}(\mathcal{B})$ is the conic hull of \mathcal{B} , and $\text{Proj}_{\mathbf{r}, \omega}(\mathcal{B})$ is the projection of the set \mathcal{B} on the coordinates $[\mathbf{r}^T, \omega^T]^T$ where $\mathbf{r} = [R_e | e \in \mathcal{E}_U]$ and $\omega = [H(Y_s) | s \in \mathcal{S}]$.

While the analytical expression determines, in principle, the rate region of any network under network coding, it is only an implicit characterization. This is because Γ_N^* is unknown and even non-polyhedral for $N \geq 4$. Further, while $\bar{\Gamma}_N^*$ is a convex cone for all N , Γ_N^* is already non-convex by $N = 3$, though it is also known that the closure only adds points at the boundary of $\bar{\Gamma}_N^*$. Thus, the direct calculation of rate regions from (1) for a network with 4 or more variables is infeasible. On a related note, at the time of writing, it appears to be unknown by the community whether or not the closure after the conic hull is actually necessary¹ in (1), and the uncertainty that necessitates its inclusion muddles a number of otherwise simple proofs and ideas. For this reason, some of the discussion in the remainder of the manuscript will study a closely related inner bound $\mathcal{R}_*(\mathbf{A})$ to $\mathcal{R}_c(\mathbf{A})$ described in the following corollary, which also introduces polyhedral inner and outer bounds through it. In all of the cases where the rate region has been computed to date $\mathcal{R}_c(\mathbf{A}) = \mathcal{R}_*(\mathbf{A})$.

Corollary 1: *The rate region $\mathcal{R}_c(\mathbf{A})$ of a network A is inner bounded by the region*

$$\mathcal{R}_*(\mathbf{A}) = \text{Proj}_{\mathbf{r}, \omega} \text{con}(\Gamma_N^*) \cap \mathcal{L}_A \quad (2)$$

which is further inner bounded, for any finite $\mathcal{A} \subset \Gamma_N^*$ of entropic vectors, by the polyhedral inner bound

$$\mathcal{R}_c(\mathbf{A}) \supseteq \mathcal{R}_*(\mathbf{A}) \supseteq \text{Proj}_{\mathbf{r}, \omega}(\text{con}(\mathcal{A}) \cap \mathcal{L}_A). \quad (3)$$

Likewise, letting Γ_N^{out} be a closed polyhedral cone that contains $\bar{\Gamma}_N^*$, then a polyhedral outer bound to the rate region is given by

$$\mathcal{R}_c(\mathbf{A}) \subseteq \text{Proj}_{\mathbf{r}, \omega}(\Gamma_N^{\text{out}} \cap \mathcal{L}_A). \quad (4)$$

Of particular interest in this manuscript are the rate region bounds $\mathcal{R}_o(\mathbf{A})$, $\mathcal{R}_{s,q}(\mathbf{A})$, $\mathcal{R}_q^{N'}(\mathbf{A})$, $\mathcal{R}_q(\mathbf{A})$, $\mathcal{R}_{\text{linear}}(\mathbf{A})$ built from the Shannon outer bound Γ_N [28], [29], scalar linear codes over the finite field \mathbb{F}_q , vector linear codes over \mathbb{F}_q of total dimension at most N' , all vector linear codes over \mathbb{F}_q , and timesharing vector linear codes over possibly different finite fields, respectively [26], [27].

$$\mathcal{R}_o(\mathbf{A}) = \text{proj}_{\mathbf{r}, \omega}(\Gamma_N \cap \mathcal{L}_A) \quad (5)$$

$$\mathcal{R}_{s,q}(\mathbf{A}) = \text{proj}_{\mathbf{r}, \omega}(\Gamma_N^q \cap \mathcal{L}_A), \quad (6)$$

$$\mathcal{R}_q^{N'}(\mathbf{A}) = \text{proj}_{\mathbf{r}, \omega}(\Gamma_{N,N'}^q \cap \mathcal{L}_A), \quad (7)$$

$$\mathcal{R}_q(\mathbf{A}) = \text{proj}_{\mathbf{r}, \omega}(\Gamma_{N,\infty}^q \cap \mathcal{L}_A), \quad (8)$$

$$\mathcal{R}_{\text{linear}}(\mathbf{A}) = \text{proj}_{\mathbf{r}, \omega}(\Gamma_N^{\text{linear}} \cap \mathcal{L}_A). \quad (9)$$

¹The closure would be unnecessary if $\bar{\Gamma}_N^* = \text{con}(\Gamma_N^*)$, i.e. if every extreme ray in $\bar{\Gamma}_N^*$ had at least one point along it that was entropic (i.e. in Γ_N^*). At present, all that is known is that Γ_N^* has a solid core, i.e. that the closure only adds points on the boundary of $\bar{\Gamma}_N^*$.

As is clear from their definition $\mathcal{R}_{s,q}(\mathbf{A}) \subseteq \mathcal{R}_q^{N'}(\mathbf{A}) \subseteq \mathcal{R}_q(\mathbf{A}) \subseteq \mathcal{R}_{\text{linear}}(\mathbf{A}) \subseteq \mathcal{R}_o(\mathbf{A})$.

Thanks to [30], Γ_N^{linear} is known for $N \leq 5$, but for $N \geq 6$, [30], [31] show that there are new inequalities for each $N - 1$ to N , and Γ_N^{linear} remains unknown. Hence, a calculation of $\mathcal{R}_{\text{linear}}(\mathbf{A})$ via the projection process in (9) is only feasible for $N \leq 5$.

The information theoretic converse prover [18], [23] can calculate $\mathcal{R}_o(\mathbf{A})$ for larger, yet still small, numbers of variables $N \leq 8$, while the information theoretic achievability prover [19], [24] can calculate $\mathcal{R}_q^{N'}(\mathbf{A})$ for small field sizes and even larger N' . As is described in the articles [18] and [19], respectively, these two pieces of software use sophisticated methods, beyond naïve calculation of the bound to Γ_N^* followed by polyhedral projection as is verbatim suggested by (5), to push the number of variables N to be as large as possible. However, these methods are still overall limited to somewhat small problems.

The remainder of this manuscript develops a structural theory that enables these tools to determine the capacity region and its bounds for problems beyond the scale they can directly reach. The first order of business, undertaken in the next section, is to identify and remove components of the network coding problem description A inessential to the core problem complexity. Subsequent sections will then set about constructing network generation tools required to build a massive database of solved network coding rate regions, organizing this database through embedding operators, and extending it to larger networks via both embedding and combinations operators.

III. REDUCTION TO A MINIMAL NETWORK

Though in principle, any network coding problem as described in §II forms a valid network coding problem, such a problem can include networks with nodes, edges, and sources which are completely extraneous and unnecessary from the standpoint of determining the rate region. To deal with this, in this section, we show how to form a network instance with equal or fewer number of sources, edges, or nodes, from an instance with extraneous components. We will show the rate region of the instance with the extraneous components can be directly inferred from the rate region of the reduced network. Network coding problems without such extraneous and unnecessary components will be called *minimal*.

Definition 1: *An acyclic network instance $\mathbf{A} = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$ is minimal if it obeys the constraints (C1)–(C14) in Table I.*

The intuition supporting each of the 14 minimality conditions is provided in Fig. 2. The key idea behind minimality is that if a network is non-minimal, we can replace it with a minimal one, determine the rate region of the minimal network, then perform simple operations on the rate of the minimal network to get the rate region of the non-minimal one. These operations are captured in Table II.

Theorem 2: *Suppose a network instance $\mathbf{A}' = (\mathcal{S}', \mathcal{G}', \mathcal{T}', \mathcal{E}', \beta')$, with rate region bounds $\mathcal{R}_l(\mathbf{A}')$, $l \in \{c, *, q, (s, q), o\}$, is not a minimal network, so that at least one of the conditions (C1)–(C14) in Def. 1 and Table I apply to \mathbf{A}' .*

An acyclic network instance $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$ is *minimal* if it obeys the following constraints:

Source minimality:

- (C1) all sources cannot be only directly connected with sinks: $\forall s \in \mathcal{S}, \text{Hd}(\text{Out}(s)) \cap \mathcal{G} \neq \emptyset$;
- (C2) sinks do not demand sources to which they are directly connected: $\forall s \in \mathcal{S}, t \in \mathcal{T}$, if $t \in \text{Hd}(\text{Out}(s))$ then $s \notin \beta(t)$;
- (C3) every source is demanded by at least one sink: $\forall s \in \mathcal{S}, \exists t \in \mathcal{T}$ such that $s \in \beta(t)$;
- (C4) sources connected to the same intermediate node and demanded by the same set of sinks should be merged: $\nexists s, s' \in \mathcal{S}$ such that $\text{Hd}(\text{Out}(s)) = \text{Hd}(\text{Out}(s'))$ and $\gamma(s) = \gamma(s')$, where $\gamma(s) = \{t \in \mathcal{T} | s \in \beta(t)\}$;

Node minimality:

- (C5) intermediate nodes with identical inputs should be merged: $\nexists k, l \in \mathcal{G}$ such that $\text{In}(k) = \text{In}(l)$;
- (C6) intermediate nodes should have nonempty inputs and outputs, and sink nodes should have nonempty inputs: $\forall g \in \mathcal{G}, t \in \mathcal{T}, \text{In}(g) \neq \emptyset, \text{Out}(g) \neq \emptyset, \text{In}(t) \neq \emptyset$;

Edge minimality:

- (C7) all hyperedges must have at least one head: $\nexists e \in \mathcal{E}$ such that $\text{Hd}(e) = \emptyset$;
- (C8) identical edges should be merged: $\nexists e, e' \in \mathcal{E}$ with $\text{Tl}(e) = \text{Tl}(e'), \text{Hd}(e) = \text{Hd}(e')$;
- (C9) intermediate nodes with unit in and out degree, and whose in edge is not a hyperedge, should be removed: $\nexists e, e' \in \mathcal{E}, g \in \mathcal{G}$ such that $\text{In}(g) = e, \text{Hd}(e) = g, \text{Out}(g) = e'$;

Sink minimality:

- (C10) there must exist a path to a sink from every source wanted by that sink: $\forall t \in \mathcal{T}, \beta(t) \subseteq \sigma(t)$, where $\sigma(t) = \{k \in \mathcal{S} | \exists \text{ a path from } k \text{ to } t\}$;
- (C11) every pair of sinks must have a distinct set of incoming edges: $\forall t, t' \in \mathcal{T}, i \neq j, \text{In}(t) \neq \text{In}(t')$;
- (C12) if one sink receives a superset of inputs of a second sink, then the two sinks should have no common sources in demand: If $\text{In}(t) \subseteq \text{In}(t')$, then $\beta(t) \cap \beta(t') = \emptyset$;
- (C13) if one sink receives a superset of inputs of a second sink, then the sink with superset input should not have direct access to the sources that demanded by the sink with subset input: If $\text{In}(t) \subseteq \text{In}(t')$ then $t' \notin \text{Hd}(\text{Out}(s))$ for all $s \in \beta(t)$.

Connectivity:

- (C14) the direct graph associated with the network A is weakly connected.

Table I: Definition of a minimal network.

Define $\mathbf{R}_{\setminus \mathcal{A}} = \text{Proj}_{\setminus \mathcal{A}} \mathbf{R}$ to be the projection of \mathbf{R} excluding coordinates associated with \mathcal{A} . For each condition (C1)–(C14) that A' violates, a simpler network A can be created according to the operations described in the middle of table II, from whose rate regions and bounds $\mathcal{R}_l(A)$, can be determined the rate regions and bounds of A' according to the relationship at the right of Table II.

Proof: See Appendix A. ■

In general, we can define a minimality operator $A = \text{minimal}(A')$ on networks, which checks the minimality conditions (C1)–(C14) on A' one by one, in the order (C1), (C2), (C6), (C5), (C3), (C4), (C7)–(C14). If any of the conditions encountered is not satisfied, the network is immediately reduced it according to the associated reduction in Theorem 2, and the resulting reduced network is checked again for minimality by starting again at condition (C1), if needed, until all minimality conditions are satisfied. Furthermore, define the associated rate region operator $\mathcal{R}_*(A') = \text{minimal}_{A' \leftarrow A}(\mathcal{R}_*(A))$ which moves through each of the reduction steps applied by $\text{minimal}(A')$ to the network A' in reverse order, utilizing the expression for the rate region change under each reduction, thereby obtaining the rate region of A' from A . Accordingly, let $\mathcal{R}_*(A) = \text{minimal}_{A' \rightarrow A}(\mathcal{R}_*(A'))$ be the rate region operator which moves through each of the reduction steps applied by $\text{minimal}(A')$ to the network A' in order, utilizing the expression for the rate region change under each reduction, thereby obtaining the rate region of A from A' . This network minimality operator and its associated rate region operators will come in use later in the paper in §VI–VIII, where it will be used to reduce to result of network combination and embedding operations on a collection of minimal networks, the result of which may or may not be minimal in general, to a minimal form. The next section sets about the problem of exhaustively generating all non-isomorphic minimal networks of a given size.

IV. ISOMORPH-FREE EXHAUSTIVE GENERATION OF MINIMAL NETWORKS

Even though the notion of network minimality (§III) reduces the set of network coding problem instances by removing parts of a network coding problem which are inessential, much more needs to be done to group network coding problem instances into appropriate equivalence classes. An important notion of equivalent problems arises from symmetries between different problems. Symmetries between different network coding problems arise because although we have to use label sets to describe the edges and sources in order to specify a network coding problem instance (identifying a certain source as source number one, another as source number two, and so on), it is clear that the essence of the underlying network coding problem is insensitive to these labels. Two networks differing only in the selection of these labels are clearly equivalent. Our goal in this section is to first formalize this notion of equivalence through relabeling, then devise an algorithm which can directly list one network coding problem representative from each equivalence class.

A. Encoding a Network Coding Problem

Though, as is consistent with the network coding literature, we have thus far utilized a tuple $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$ to represent a network instance, an alternative more concise representation of a network coding problem is more amenable to recognizing minimality and equivalence through relabeling.

In particular, a network instance with K sources and L edges that obeys the minimality conditions (C1–C14) can be encoded as an ordered pair $(\mathcal{Q}, \mathcal{W})$ consisting of a set \mathcal{Q} of edge definitions $\mathcal{Q} \subseteq \{(i, \mathcal{A}) | i \in \{K+1, \dots, K+L\}, \mathcal{A} \subseteq \{1, \dots, K+L\} \setminus \{i\}, |\mathcal{A}| > 0\}$, and a set \mathcal{W} of sink definitions $\mathcal{W} \subseteq \{(i, \mathcal{A}) | i \in \{1, \dots, K\}, \mathcal{A} \subseteq \{1, \dots, K+L\} \setminus \{i\}\}$. Here, the sources are associated with labels $\{1, \dots, K\}$ and the edges are associated with labels $\{K+1, \dots, K+L\}$.

#	Redundant Part	How to construct A	Rate Region Relationship
(D1)	$\exists s' \in \mathcal{S}'$ such that $\text{Hd}(\text{Out}(s')) \cap \mathcal{G}' = \emptyset$	A will be A' with s' removed	if $\exists t' \in \mathcal{T}'$ such that $s \in \beta(t')$ and $s \notin \text{In}(t')$: $\mathcal{R}_l(A') := \{\mathbf{R} \mathbf{R}_{\setminus s'} \in \mathcal{R}_l(A), \omega_{s'} = 0\}$. Otherwise: $\mathcal{R}_l(A') := \{\mathbf{R} \mathbf{R}_{\setminus s} \in \mathcal{R}_l(A), \omega_s \geq 0\}$. $\mathcal{R}_l(A) = \text{Proj}_{\setminus s'} \mathcal{R}_l(A')$.
(D2)	$\exists s' \in \mathcal{S}', t' \in \mathcal{T}'$, such that $t' \in \text{Hd}(\text{Out}(s'))$ and $s' \in \beta(t')$	A will be A' with $\text{In}(t) = \text{In}(t') \setminus s'$ and $\beta(t) = \beta(t') \setminus s'$	$\mathcal{R}_l(A') = \mathcal{R}_l(A)$
(D3)	$\exists s' \in \mathcal{S}'$, such that $\forall t' \in \mathcal{T}', Y_{s'} \notin \beta(t')$	A will be A' with removal of the redundant source s'	$\mathcal{R}_l(A') := \{\mathbf{R} \mathbf{R}_{\setminus s'} \in \mathcal{R}_l(A), H(Y_{s'}) \geq 0\}$ $\mathcal{R}_l(A) = \text{Proj}_{\setminus s'} \mathcal{R}_l(A')$
(D4)	$\exists s, s' \in \mathcal{S}'$ such that $\text{Hd}(\text{Out}(s)) = \text{Hd}(\text{Out}(s'))$ and $\gamma(s) = \gamma(s')$	A will be A' with sources s, s' merged	$\forall l \in \{c, *, q, o\}$: $\mathcal{R}_l(A') = \{\mathbf{R} [\mathbf{R}_{\setminus \{s, s'\}}^T, H(Y_s) + H(Y_{s'})]^T \in \mathcal{R}_l(A)\}$ i.e., replace $H(Y_s)$ in $\mathcal{R}_l(A)$ with $H(Y_s) + H(Y_{s'})$ to get $\mathcal{R}_l(A')$ $\mathcal{R}_l(A) = \{\mathbf{R}_{\setminus \{s\}} \mathbf{R} \in \mathcal{R}_l(A'), \omega_{s'} = 0\}$
(D5)	$\exists k', j' \in \mathcal{G}$ such that $\text{In}(k') = \text{In}(j')$	A will be A' with k', j' merged so that $\text{In}(k) = \text{In}(k') = \text{In}(j')$, $\text{Out}(k) = \text{Out}(k') \cup \text{Out}(j')$, and $\mathcal{G} = \mathcal{G}' \setminus j'$	$\mathcal{R}_l(A) = \mathcal{R}_l(A')$
(D6)	$\exists g' \in \mathcal{G}$ such that $\text{In}(g) = \emptyset$, or $\text{Out}(g) = \emptyset$	A will be A' with removal of the redundant node(s) g'	$\mathcal{R}_l(A') = \mathcal{R}_l(A)$
(D6)	$\exists t' \in \mathcal{T}$ such that $\text{In}(t') = \emptyset$	A will be A' with removal of the redundant node(s) t' and the deletion of any sources it demands	$\mathcal{R}_l(A') = \{\mathbf{R} \mathbf{R}_{\setminus \beta(t')} \in \mathcal{R}_l(A), \omega_s = 0 \forall s \in \beta(t')\}$ $\mathcal{R}_l(A) = \text{Proj}_{\setminus \beta(t')} \mathcal{R}_l(A')$
(D7)	$\exists e' \in \mathcal{E}'$ such that $\text{Hd}(e') = \emptyset$	A will be A' with removal of edge e'	$\mathcal{R}_l(A') = \{\mathbf{R} \mathbf{R}_{\setminus e'} \in \mathcal{R}_l(A), R_{e'} \geq 0\}$ $\mathcal{R}_l(A) = \text{Proj}_{\setminus e'} \mathcal{R}_l(A')$
(D8)	$\exists e, e' \in \mathcal{E}'$ with $\text{Ti}(e) = \text{Ti}(e')$, $\text{Hd}(e) = \text{Hd}(e')$	A will be A' with edges e, e' merged as e	$\mathcal{R}_l(A') = \{\mathbf{R} [\mathbf{R}_{\setminus \{e, e'\}}^T, R_e + R_{e'}]^T \in \mathcal{R}_l(A)\}$ i.e., replace R_e in $\mathcal{R}_*(A)$ with $R_e + R_{e'}$ to get $\mathcal{R}_*(A')$. $\mathcal{R}_l(A) = \{\mathbf{R}_{\setminus \{e, e'\}}^T \mathbf{R} \in \mathcal{R}_l(A'), R_{e'} = 0\}$
(D9)	$\exists e, e' \in \mathcal{E}', g' \in \mathcal{G}'$ such that $\text{In}(g') = e$, $\text{Hd}(e) = g'$, $\text{Out}(g') = e'$	A will be A' with the node g' removed and a new edge e_i replacing e, e' by directly connecting $\text{Ti}(e)$ and $\text{Hd}(e')$	$\mathcal{R}_l(A') = \{\mathbf{R} [\mathbf{R}_{\setminus \{e, e'\}}^T, \min\{R_e, R_{e'}\}]^T \in \mathcal{R}_l(A)\}$ i.e., replace R_e in $\mathcal{R}_l(A)$ with $\min\{R_e, R_{e'}\}$ to get $\mathcal{R}_l(A')$. $\mathcal{R}_l(A) = \{\mathbf{R}_{\setminus \{e, e'\}}^T \mathbf{R} \in \mathcal{R}_l(A')\}$.
(D10)	$\exists t' \in \mathcal{T}', s' \in \mathcal{S}'$, such that $s' \in \beta(t')$ but $s' \notin \sigma(t')$	A will be A' with s' deleted	$\mathcal{R}_l(A') = \{\mathbf{R} \mathbf{R}_{\setminus s'} \in \mathcal{R}_l(A), H(Y_{s'}) = 0\}$ $\mathcal{R}_l(A) = \text{Proj}_{\setminus s'} \mathcal{R}_l(A')$
(D11)	$\exists t, t' \in \mathcal{T}', t \neq t'$, such that $\text{In}(t) = \text{In}(t')$	A will be A' with sinks t, t' merged	$\mathcal{R}_l(A') = \mathcal{R}_l(A)$
(D12)	$\exists t, t'$ such that $\text{In}(t) \subseteq \text{In}(t')$ and $\beta(t) \cap \beta(t') \neq \emptyset$	A will be A' with removal of $\beta(t) \cap \beta(t')$ from $\beta(t')$	$\mathcal{R}_l(A') = \mathcal{R}_l(A)$
(D13)	$\exists t, t', s' \in \beta(t)$ such that $\text{In}(t) \subseteq \text{In}(t')$ and $t' \in \text{Hd}(\text{Out}(s'))$	A will be A' with removal of s' from $\text{In}(t')$	$\mathcal{R}_l(A') = \mathcal{R}_l(A)$
(D14)	A' is not weakly connected	A_1, A_2 are two weakly disconnected components	$\mathcal{R}_l(A') = \mathcal{R}_l(A_1) \times \mathcal{R}_l(A_2)$

Table II: Relationship between the rate region of the non-minimal network A' and A for all $l \in \{c, *, q, (s, q), o\}$ unless otherwise noted.

Each $(i, \mathcal{A}) \in \mathcal{Q}$ indicates that the edge $i \in \mathcal{E}_U$ is encoded exclusively from the sources and edges in \mathcal{A} , and hence represents the information that $\mathcal{A} = \text{In}(\text{Ti}(i))$. Furthermore, each sink definition $(i, \mathcal{A}) \in \mathcal{W}$ represents the information that there is a sink node whose inputs are \mathcal{A} and which decodes source i as its output. Note that there are L non-source edges in the network, each of which must have some input according to condition (C6). We additionally have the requirement that $|\mathcal{Q}| = L$, and, to ensure that no edge is multiply defined, we must have that if (i, \mathcal{A}) and (i', \mathcal{A}') are two different elements in \mathcal{Q} , then $i \neq i'$. As the same source may be decoded at multiple sinks, there is no such requirement for \mathcal{W} .

A key characteristic of the representation $(\mathcal{Q}, \mathcal{W})$ is that a network encoded this way is guaranteed to obey several of the key minimality constraints, including (C5), (C11), and (C8).

B. Expressing Network Equivalence with a Group Action

Another benefit of the representation of the network coding problem as the ordered pair $(\mathcal{Q}, \mathcal{W})$ is that it enables the notion of network isomorphism to be appropriately defined. In particular, let $\mathbf{G} := S_{\{1, 2, \dots, K\}} \times S_{\{K+1, \dots, K+L\}}$ be the direct product of the symmetric group of all permutations of the set $\{1, 2, \dots, K\}$ of source indices and the symmetric group of all permutations of the set $\{K+1, \dots, K+L\}$ of edge indices. The group \mathbf{G} acts in a natural manner on the elements of the sets \mathcal{Q}, \mathcal{W} of edge and sink definitions. In particular, let $\pi \in \mathbf{G}$ be a permutation in \mathbf{G} , then the group action maps

$$\pi((i, \mathcal{A})) \mapsto (\pi(i), \pi(\mathcal{A})) \quad (10)$$

with the usual interpretation that $\pi(\mathcal{A}) = \{\pi(j) | j \in \mathcal{A}\}$. This extends to action on the sets \mathcal{Q} and \mathcal{W} by acting on each element of the set, and thus to the ordered pair $(\mathcal{Q}, \mathcal{W})$, by acting on the two sets in the pair, as well.

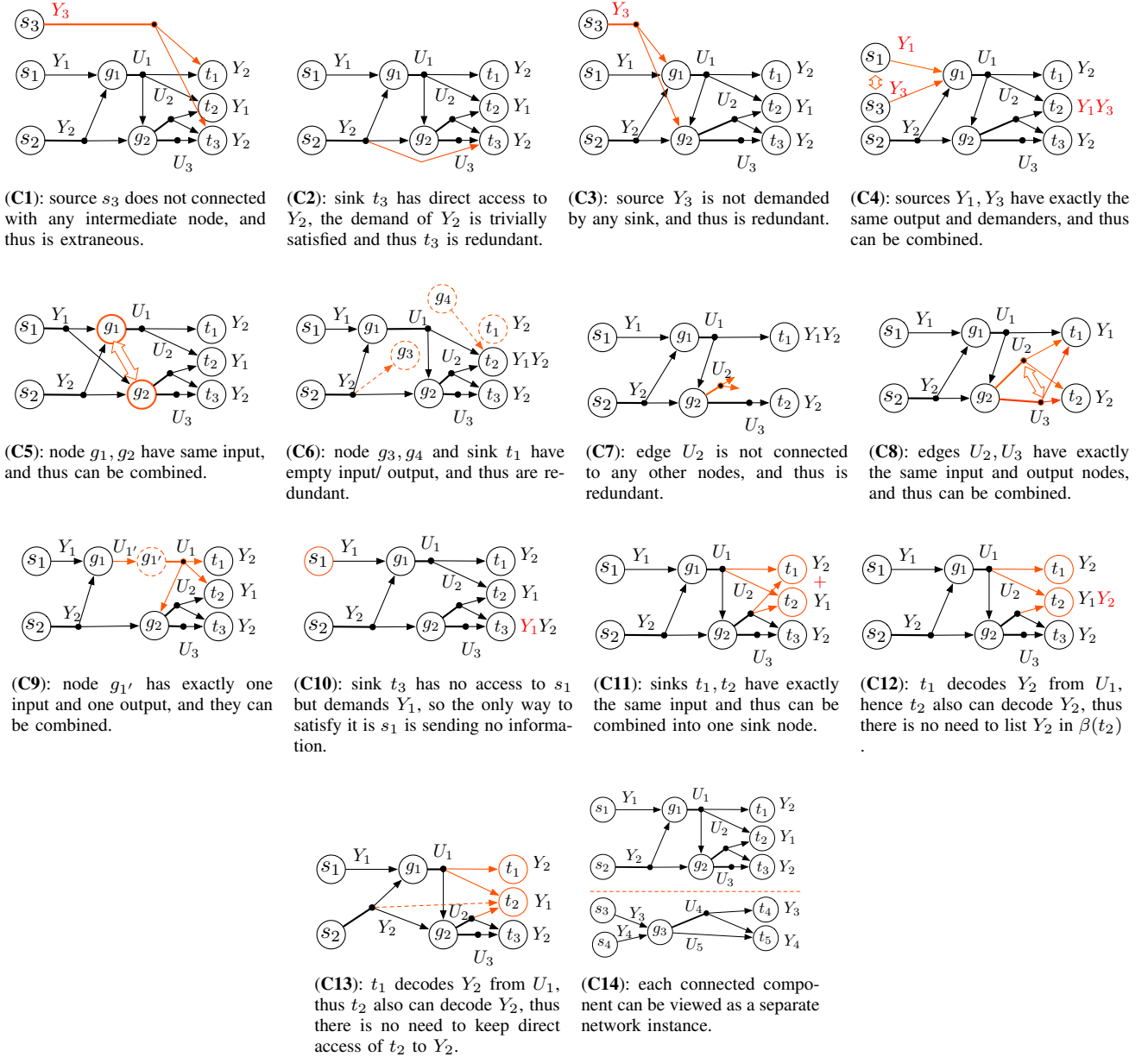


Figure 2: Examples to demonstrate the minimality conditions (C1)–(C14).

Two networks (Q_1, \mathcal{W}_1) and (Q_2, \mathcal{W}_2) are said to be *isomorphic*, or in the same equivalence class, if there is some permutation of $\pi \in \mathbf{G}$ such that $\pi((Q_1, \mathcal{W}_1)) = (Q_2, \mathcal{W}_2)$. The equivalence classes are called orbits in the language of group actions.

We elect to represent each equivalence class with its *canonical network*, which is the element in each orbit that is least in a lexicographic sense. Note that this lexicographic (i.e., dictionary) order is well-defined, as we can compare two subsets \mathcal{A} and \mathcal{A}' by viewing their members in increasing order (under the usual ordering of the integers $\{1, \dots, L+K\}$) and lexicographically comparing them. This then implies that we can lexicographically order the ordered pairs (i, \mathcal{A}) according to $(i, \mathcal{A}) > (j, \mathcal{A}')$ if $j < i$ or $i = j$ and $\mathcal{A}' < \mathcal{A}$ under this lexicographic ordering. Since the elements of \mathcal{Q} and \mathcal{W}

are of the form (i, \mathcal{A}) , this in turn means that they can be ordered in increasing order, and then also lexicographically compared, enabling comparison of two edge definition sets \mathcal{Q} and \mathcal{Q}' or two sink definition sets \mathcal{W} and \mathcal{W}' . Finally, one can then use these orderings to define the lexicographic order on the network ordered pairs (Q, \mathcal{W}) . The element in an orbit $\mathcal{O}_{(Q, \mathcal{W})}$ which is minimal under this lexicographic ordering will be the canonical representative for the orbit.

A key basic result in the theory of group actions, the *Orbit Stabilizer Theorem*, states that the number of elements in an orbit is equal to the ratio of the size of the acting group \mathbf{G} and its stabilizer subgroup $\mathbf{G}_{(Q, \mathcal{W})}$ of any element selected from

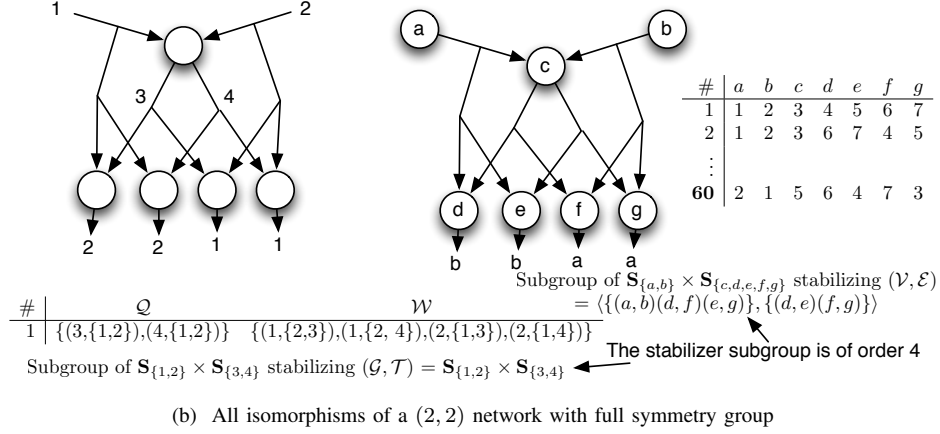
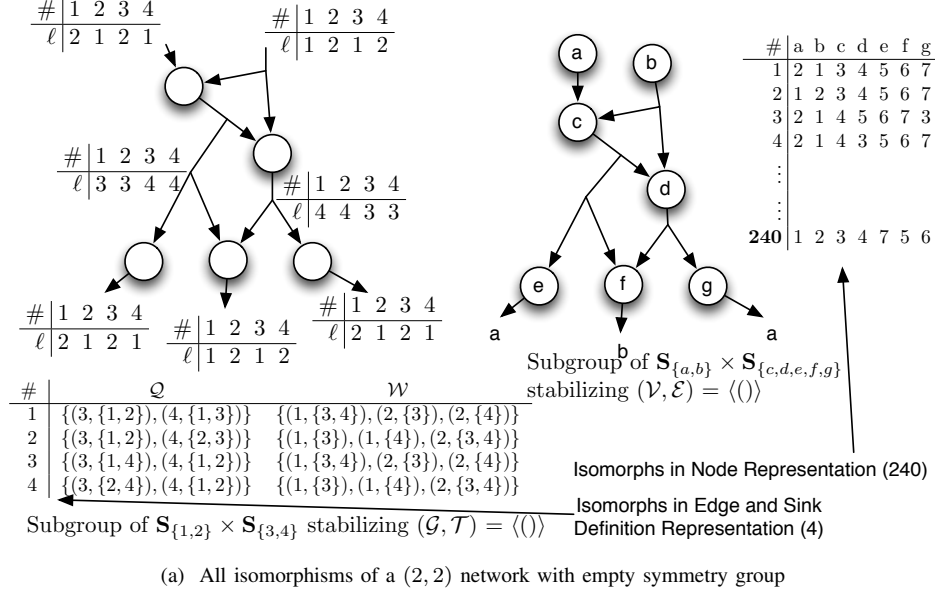


Figure 3: Examples of (2,2) networks with all edge isomorphisms (left) and all node isomorphisms (right). The instance indices are marked by # and the labels are marked by l .

the orbit:

$$|\{\pi((\mathcal{Q}, \mathcal{W})) \mid \pi \in \mathbf{G}\}| = |\mathcal{O}_{(\mathcal{Q}, \mathcal{W})}| = \frac{|\mathbf{G}|}{|\mathbf{G}_{(\mathcal{Q}, \mathcal{W})}|},$$

$$\mathbf{G}_{(\mathcal{Q}, \mathcal{W})} := \{\pi \in \mathbf{G} \mid \pi((\mathcal{Q}, \mathcal{W})) = (\mathcal{Q}, \mathcal{W})\}$$

Note that, because it leaves the sets of edges, decoder demands, and topology constraints set-wise invariant, the elements of the stabilizer subgroup $\mathbf{G}_{(\mathcal{Q}, \mathcal{W})}$ also leave the set of rate region constraints \mathcal{L}_A invariant. Such a group of permutations on sources and edges is called the *network symmetry group* [32], [33], which is further exploited in [18] and [23] to substantially reduce the complexity of computing the network's rate regions. This network symmetry group plays a role in the present study because, as depicted in Figures 3a and 3b, by the orbit stabilizer theorem mentioned above, it determines the number of networks equivalent to a given canonical network (the representative we will select from the orbit). Fig. 3a and 3b demonstrate the number of elements in the orbit of a network with a trivial network symmetry group and with a network symmetry group of order 4, respectively.

Additionally, it shows that the number of relabelings of the node representation of these two graphs is far larger than the number of relabelings of the edge representation $(\mathcal{Q}, \mathcal{W})$.

C. Network Enumeration/Listing Algorithm

Formalizing the notion of a canonical network via group actions on the set of minimal $(\mathcal{Q}, \mathcal{W})$ pairs enables one to partly develop a method for directly listing canonical networks based on techniques from computational group theory.

To solve this problem we can harness the algorithm *Leiterspiel*, loosely translated *snakes and ladders* [34], [35], which, given an algorithm for computing canonical representatives of orbits, i.e., transversal, on some finite set \mathcal{X} under a group \mathbf{G} and its subgroups, provides a method for computing the orbits on the power set $\mathcal{P}_i(\mathcal{X}) = \{\mathcal{B} \subseteq \mathcal{X} \mid |\mathcal{B}| = i\}$ of subsets from \mathcal{X} of cardinality i , incrementally in i . In fact, the algorithm can also list directly only those canonical representatives of orbits for which some test function f returns 1, provided that the test function has the property that any subset of a set with

$f = 1$ also has $f = 1$. This test function is useful for only listing those subsets in $\mathcal{P}_i(\mathcal{X})$ with a desired set of properties, provided these properties are inherited by subsets of a superset with that property.

To see how to apply and modify Leiterspiel for network coding problem enumeration, let \mathcal{X} be the set of possible edge definitions

$$\mathcal{X} := \left\{ (i, \mathcal{A}) \mid \begin{array}{l} i \in \{K+1, \dots, K+L\}, \\ \mathcal{A} \subseteq \{1, \dots, K+L\} \setminus \{i\} \end{array} \right\} \quad (11)$$

For small to moderately sized networks, the orbits in \mathcal{X} from \mathbf{G} and its subgroups can be readily computed with modern computational group theory packages such as GAP [36] or PERMLIB [37]. Leiterspiel can be applied to first calculate the non-isomorphic candidates for the edge definition set \mathcal{Q} , as it is a subset of \mathcal{X} with cardinality L obeying certain conditions associated with the definition of a network coding problem and its minimality (c.f. **C1–C14**). Next, for each non-isomorphic edge-definition \mathcal{Q} , a list of non-isomorphic sink-definitions \mathcal{A} , also constrained to obey problem definition and minimality conditions (**C1–C14**), can be created with a second application of Leiterspiel. The pseudo-code for the resulting isomorph free exhaustive generation of all network coding problems is provided in Alg. 1, a more detailed discussion of which is available in [26], [27], and an implementation of which in GAP is available at [38].

An additional pleasant side effect of Alg. 1 is that the stabilizer subgroups, i.e., the network symmetry groups [32], are directly provided by the second Leiterspiel. Harnessing these network symmetry groups provides a powerful technique to reduce the complex process of calculating the rate region for a network coding problem instance [18], [23], [33].

Although this method directly generates the canonical representatives from the network coding problem equivalence classes without ever listing other isomorphs within these classes, one can also use the stabilizer subgroups provided by Leiterspiel to directly enumerate the sizes of these equivalence classes of $(\mathcal{Q}, \mathcal{W})$ pairs, as described above via the orbit stabilizer theorem. Experiments summarized in Table III show that the number of isomorphic cases is substantially larger than the number of canonical representatives/equivalence classes, and hence the extra effort to directly list only canonical networks is worthwhile. It is also worth noting that a node representation, utilizing a node based description of the hyper edges, would yield a substantially higher number of isomorphs.

D. Enumeration Results for Networks with Different Sizes

By using our enumeration tool with an implementation of the algorithms above, we obtained the list of canonical minimal network instances for different network coding problem sizes with $N = K + L \leq 5$. While the whole list is available in a companion dataset [39], we give the numbers of network problem instances in Table III, where $|\mathcal{Z}|$, $|\hat{\mathcal{Z}}|$, $|\hat{\mathcal{Z}}_n|$ represent the number of canonical network coding problems (i.e., the number of equivalence classes), the number of edge descriptions of network coding problems including symmetries/equivalences, and the number of node descriptions of network coding problems including the symmetries/equivalences,

Input: number of sources K , number of non-source edges L
Output: All non-isomorphic network instances \mathcal{Z}
Initialization: $\mathcal{Z} = \emptyset$;
 Let $\mathcal{X} := \left\{ (i, \mathcal{A}) \mid \begin{array}{l} i \in \{K+1, \dots, K+L\}, \\ \mathcal{A} \subseteq \{1, \dots, K+L\} \setminus \{i\} \end{array} \right\}$;
 Let f_1 be the condition that $\nexists (i, \mathcal{A}), (i', \mathcal{A}')$ such that $i = i'$; Let f_2 be the condition of acyclicity;
 Let acting group $\mathbf{G} := \mathbf{S}_{\{1, \dots, K\}} \times \mathbf{S}_{\{K+1, \dots, K+L\}}$;
 Call Leiterspiel algorithm to incrementally get all candidate transversal up to L :
 $T_L = \text{Leiterspiel}(\mathbf{G}, \mathcal{P}_L^{f_1, f_2}(\mathcal{X}))$;
for each $\mathcal{Q} \in T_L$ **do**
 if \mathcal{Q} obeys (**C1**) **then**
 Let $\mathcal{X}' := \{(i, \mathcal{A}) \mid i \in \{1, \dots, K\}, \mathcal{A} \subseteq \{1, \dots, K+L\} \setminus \{i\}, \exists \text{ a directed path in } \mathcal{Q} \text{ from } i \text{ to at least one edge in } \mathcal{A}\}$;
 Let f'_1 be the condition (**C12**); Let f'_2 be the condition (**C13**);
 Let acting group $\mathbf{G} := \mathbf{S}_{\{1, \dots, K\}} \times \mathbf{S}_{\{K+1, \dots, K+L\}}$;
 Call Leiterspiel algorithm to incrementally get all candidate canonical representatives, i.e., transversals, up to no new element can be added obeying (**C12, C13**):
 $T_K = \text{Leiterspiel}(\mathbf{G}, \mathcal{P}_K^{f'_1, f'_2}(\mathcal{X}'))$;
 for each $\mathcal{W} \in T_K$ **do**
 if $(\mathcal{Q}, \mathcal{W})$ obeys (**C3–C7**) and (**C14**) **then**
 $\mathcal{Z} = \mathcal{Z} \cup (\mathcal{Q}, \mathcal{W})$;
 end
 end
 end
end

Algorithm 1: Isomorph-free Exhaustive Generation of (K, L) networks.

respectively. As we can see from the table, the number of possibilities in the node representation of the network coding problems explodes very quickly, with the more than 2 trillion labeled node network coding problems covered by the study only necessitating a list of consisting of roughly 750,000 equivalence classes of network coding problems. That said, it is also important to note that the number of non-isomorphic network instances increases exponentially fast as network size grows. For instance, the number of non-isomorphic general network instances grows from 333 to 485,890 (roughly, an increase of about 1500 times), when the network size grows from $(2, 2)$ to $(2, 3)$.

V. RATE REGIONS OF ALL MINIMAL NETWORKS OF SIZE $N = K + L \leq 5$

It is clear from the sheer scale of the numbers in Table III that there is incredible combinatorial explosion of diversity among even the smallest network coding problems, i.e. those with $K+L \leq 5$. Even after applying minimality considerations and removing symmetric problem instances, there still remain

Table III: Numbers of minimal network coding problems of different sizes: $|\mathcal{Z}|$ represents the number of equivalence classes (under relabeling) of minimal network coding problems, $|\hat{\mathcal{Z}}|$ the number of labeled minimal network coding problems in the $(\mathcal{Q}, \mathcal{W})$ edge-based representation, and $|\hat{\mathcal{Z}}_n|$ number of labeled minimal network coding problems in the $(\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$ node based representation.

(K, L)	$ \mathcal{Z} $	$ \hat{\mathcal{Z}} $	$ \hat{\mathcal{Z}}_n $
(1,2)	4	7	39
(1,3)	132	749	18 401
(1,4)	18 027	420948	600 067 643
(2,1)	1	1	6
(2,2)	333	1 270	163 800
(2,3)	485 890	5 787 074	2 204 574 267 764
(3,1)	9	31	582
(3,2)	239 187	2 829 932	176 437 964 418
(4,1)	536	10 478	12 149 472
Total	744 119	9 050 490	2 381 624 632 119

744,119 problem equivalence classes. It should be quite clear from this scale that, despite the small number of variables at play $K + L \leq 5$, this is beyond the scale that even a highly trained human could be trusted to accurately catalog or interpret with traditional techniques and without the aid of a computer. Furthermore, tackling solving the rate regions for this each of these networks in the traditional manner, even with the help of computer assisted inequality verification tools such as ITIP/xITIP/minITIP [40]–[42], would involve an immense effort, given that proving even one region by hand in this typical manner is lengthy process.

However, the recently developed rate region *generation* algorithms and software [18], [19], [23], [24] can very rapidly determine the rate regions (5)–(9) for all of these 744,119 networks. These rate regions, which are far to numerous and complicated to be individually discussed here, are available as a companion dataset [39]. This dataset includes, for each minimal network coding problem equivalence class, a converse proof for each inequality in its rate region, and a code construction for each extreme ray. Furthermore, the tightness of various inner bounds formed by various classes of codes are investigated in Table IV. Quite a bit can be learned about larger networks, with $N > 5$, from this database of rate regions using the hierarchical theory developed in the remaining sections of this manuscript, however the remainder of this particular section is devoted to reporting summary results about this dataset.

Indeed, there are natural questions about the rate regions of this class of small networks $N = K + L \leq 5$ that arise from the implicit result of [15], [17], its extension in Thm. 1 to the present context, and what is known about $\bar{\Gamma}_N^*$ for $N \leq 5$, despite the fact that it remains incompletely characterized for $N \geq 4$. First of all, it has known for quite some time [29], [43] that already at $N = 4$ there are non-Shannon inequalities, so that $\bar{\Gamma}_4^* \subsetneq \Gamma_4$, and, through results regarding scalar capacities of networks [44], that non-Shannon inequalities can thus influence capacity regions. However, given that the class of minimal networks has not been even defined, let alone catalogued until this paper, and a nearly non-existent literature on fully calculated rate regions (rather than scalar capacities), it remains open at what network size there is first a minimal network coding problem requiring the use of

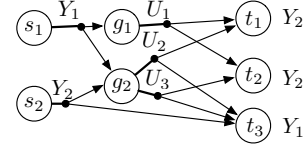


Figure 4: Block diagram of the $(2, 3)$ network A in Example 1.

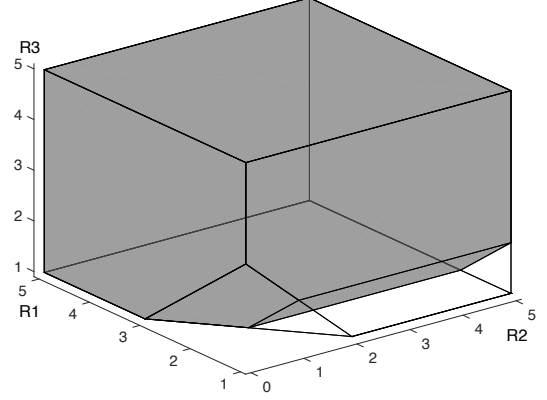


Figure 5: Comparison of rate regions $\mathcal{R}_*(A) = \mathcal{R}_o(A)$ and $\mathcal{R}_{s,2}^6(A) = \mathcal{R}_2^6(A)$, when source entropies are $(H(Y_1) = 2, H(Y_2) = 1)$ and $0 \geq R_i \leq 5, i = 1, 2, 3$. The ratio of $\mathcal{R}_2^6(A)$ over $\mathcal{R}_*(A)$ is about 97.67%.

non-Shannon inequalities. Indeed, even though $\bar{\Gamma}_4^* \subsetneq \Gamma_4$ and $\bar{\Gamma}_5^* \subsetneq \Gamma_5$, and even though for every non-Shannon inequality there exists a network whose capacity region depends on it [4], [6], [7], the question arises, is the phenomenon of *rate regions* (not scalar capacities) depending on non-Shannon inequalities already visible with networks $N = 4, 5$? Thus far, examples necessitating non-Shannon inequalities involve rate regions which remain unknown and far larger numbers of variables [44], [45].

Another important question recognizes that linear codes, even time shared across mixed field sizes and characteristics, exhaust only part of the entropy region, so that $\Gamma_N^{\text{linear}} \subsetneq \bar{\Gamma}_N^*$ for $N \geq 4$. Furthermore, unlike $\bar{\Gamma}_N^*$, Γ_N^{linear} is completely known for $N = 4, 5$ [46], [47]. Despite this fact, the famous example showing the necessity of nonlinear codes in network coding [48] by pasting the Fano and non-Fano matroids together in a manner consistent with the example displaying an algebraic but not linearly representable matroid in Ingleton's paper [49], utilizes a large network. Given that $\Gamma_N^{\text{linear}} \subsetneq \bar{\Gamma}_N^*$ already for $N \in \{4, 5\}$ are there networks of this smaller size necessitating nonlinear codes?

Table IV answers the questions in both of the two previous paragraphs in the negative – linear codes and Shannon type inequalities suffice for (even hypergraph) networks built from $N = K + L \leq 5$ random variables.

To give a sense of the type of complexity that does already occur in this class of small networks, the next example discusses the capacity regions of a representative element of the database [39].

Example 1: A 2-source 3-encoder hyperedge network instance A with block diagram shown in Fig. 4, for which the rate region

Table IV: Sufficiency of codes for network instances: Columns 3–8 show the number of instances that the rate region inner bounds match with the Shannon outer bound.

(K, L)	$ \mathcal{Z} $	$\mathcal{R}_{s,2}(\mathbf{A})$	$\mathcal{R}_2^{N+1}(\mathbf{A})$	$\mathcal{R}_2^{N+2}(\mathbf{A})$	$\mathcal{R}_2^{N+3}(\mathbf{A})$	$\mathcal{R}_2^{N+4}(\mathbf{A})$	$\mathcal{R}_{\text{linear}}^N$
(1, 2)	4	4	4	4	4	4	4
(1, 3)	132	122	132	132	132	132	132
(1, 4)	18027	13386	16930	17697	17928	17928	18027
(2, 1)	1	1	1	1	1	1	1
(2, 2)	333	301	319	323	323	333	333
(2, 3)	485890	341406	403883	432872	434545	435671	485890
(3, 1)	9	4	4	9	9	9	9
(3, 2)	239187	118133	168761	202130	211417	214171	239187
(4, 1)	536	99	230	235	476	476	536
Total:	744119	473456	590264	653403	664835	668725	744119

$\mathcal{R}_*(\mathbf{A}) = \mathcal{R}_c(\mathbf{A})$ is described the inequalities

$$R_1 + R_2 + R_3 \geq H(Y_1) + 2H(Y_2) \quad (12)$$

$$R_2 \geq H(Y_2) \quad (13)$$

$$R_3 \geq H_2 \quad (14)$$

$$R_2 + R_3 \geq H(Y_1). \quad (15)$$

Scalar binary codes do not suffice for this network, as the rate region reachable with scalar binary codes is $\mathcal{R}_{s,2}(\mathbf{A})$ is given by the inequalities

$$R_1 + R_2 + R_3 \geq H(Y_1) + 2H(Y_2) \quad (16)$$

$$R_2 \geq H(Y_2) \quad (17)$$

$$R_3 \geq H_2 \quad (18)$$

$$R_2 + R_3 \geq H(Y_1) + H(Y_2). \quad (19)$$

The regions $\mathcal{R}_{s,2}(\mathbf{A})$ and $\mathcal{R}_*(\mathbf{A}) = \mathcal{R}_c(\mathbf{A}) = \mathcal{R}_o(\mathbf{A})$ differ owing to the extreme ray $[R_1, R_2, R_3, H(Y_1), H(Y_2)] = [2, 1, 1, 2, 1]$ of $\mathcal{R}_o(\mathbf{A})$. This extreme ray cannot be achieved by scalar binary codes (as is evident from the inclusion of rates of both 1 and 2 in the ray), but can be achieved with a vector binary code $U_1 = [Y_1^1, Y_1^2], U_2 = Y_1^1 + Y_2, U_3 = Y_1^2 + Y_2$ with Y_1^1, Y_1^2 being the two bits in Y_1 . Note that from the achieving code, we also see that this extreme ray cannot be achieved by binary codes from 6 bits because at least 7 bits are necessary ($2+1+1+2+1=7$). Indeed, for this example, we have $\mathcal{R}_2^6(\mathbf{A}) = \mathcal{R}_{s,2}(\mathbf{A})$ and $\mathcal{R}_2^7(\mathbf{A}) = \mathcal{R}_*(\mathbf{A})$. This is illustrated in Fig. 5 by choosing a particular source entropy tuple. When source entropies are $[H(Y_1), H(Y_2)] = [2, 1]$ and the cone is capped by $R_i \leq 5, i = 1, 2, 3$, there is a clear gap between the two polytopes. The scalar inner bound occupies about 97.67% of the volume of the actual rate region for this choice of $[H(Y_1), H(Y_2)]$.

Harnessing greater computational power and more efficient implementations, the algorithm in §IV could be utilized to generate even larger minimal network coding rate region databases than the one provided with this article [39], yet the primary issues with this approach of exhaustively cataloguing of network coding capacity regions are already evident with this example database. In particular, the natural question arises as to how one can really learn from such a massive database, aside from answering particular existence queries such as the previous two, or searching for counter-examples to putative theorems. Network coding problems arising in applications

will often be larger, and the key issue is what can be learned from a database of small network coding problems about them.

It is with this aim, of learning properties of larger network coding problems from such an exhaustive database of small network coding rate regions, that we presently shift focus in the next two sections to developing new theoretical tools. Our effort in this direction aims to provide organizing principles linking network coding problems of different sizes, enabling some of the combinatorial explosion encountered as the network size grows to be handled through hierarchy. In particular, §VIII will demonstrate that the notions of hierarchy developed in §VI and §VII enable any exhaustive database of rate regions of small networks to be leveraged to 1) determine cases when the capacity region of a network of arbitrary scale can not be exhausted by a particular class of codes, and 2) using the small networks as building blocks, construct rate regions of networks of arbitrary scale.

VI. NETWORK EMBEDDING OPERATIONS

In this section, we set about formalizing a manner in which a smaller network coding problem can be recognized as being embedded in a larger one. While other articles have also considered reductions shrinking network coding problems [50], [51], of chief interest here will be recognizing those notions of embeddings such that rate region properties, such as sufficiency of a class of linear codes to exhaust the capacity region, or tightness of the Shannon outer bound, are inherited by the smaller network from the larger one. In this manner, we will be able to explain insufficiency of a class of codes, or lack of tightness of the Shannon outer bound, for a particular large network, to be boiled down to or explained by the inclusion of a smaller network, a forbidden network minor for this property, within it.

This approach draws direct inspiration from the celebrated well-quasi-ordering result of graph theory, the Robertson-Seymour theorem [52], [53], which states that any minor closed family of graphs must have at most a finite number of forbidden minors. Network coding capacity regions build upon polymatroids, which build upon matroids, which build upon graphs. Matroids do not exhibit well quasi-ordering under matroid minors, however families of matroids representable over certain finite fields can be characterized by finite lists of forbidden minors [54], [55], and the recently claimed [56] Rota's conjecture states that this is true for any finite field.

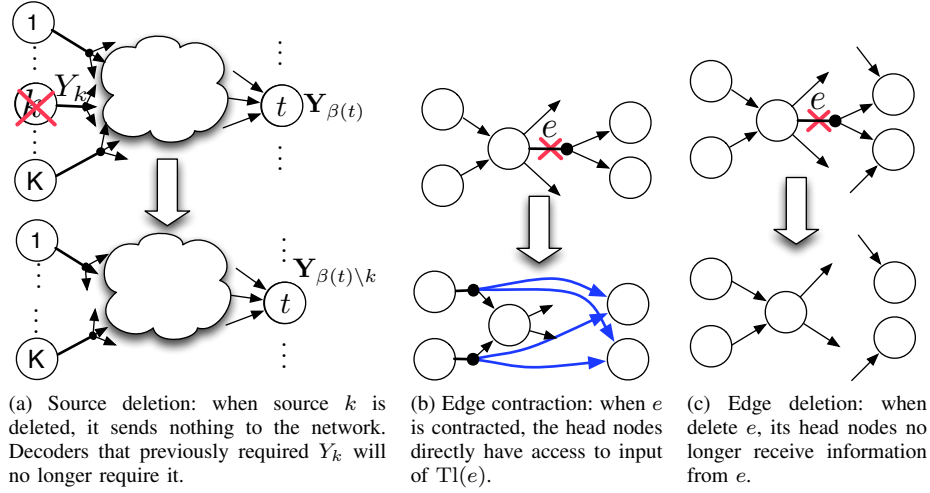


Figure 6: Definitions of embedding operations on a network

Recently, [57] presented a class of embedding operations specially constructed for a class of network coding problems, multilevel diversity coding systems, in which all sources are available at all encoders, sources must be decoded in a prioritized order at sinks, and there are no intermediate nodes. These operators were shown to have the desired inheritance properties for linear code class sufficiency and tightness of the Shannon outer bound, for the associated rate regions [57], however, as they were constructed for the special highly restricted class of MDCS networks, they enforce additional constraints from that context which are unnecessary in the present one. Passing to the present context of general network coding problems, much more natural notions of network coding problem embedding that more directly mimic graph minors, which are exclusively built from edge deletion and contraction, can be developed as depicted in Fig. 6 and in the following definitions.

Definition 2 (Source Deletion ($A \setminus k$)): The result of deleting source $k \in \mathcal{S}$ from $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$, is $A'' = \text{minimal}(A')$, where $A' = A \setminus k = (\mathcal{S}', \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta')$ with $\mathcal{S}' = \mathcal{S} \setminus k$ and $\beta' = (\beta(t) \setminus k, t \in \mathcal{T})$.

Definition 3 (Edge Deletion ($A \setminus e$)): The result of deleting edge $e \in \mathcal{E}$ from $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$, is $A'' = \text{minimal}(A')$, wherein $A' = A \setminus e = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}', \beta)$ with $\mathcal{E}' = \mathcal{E} \setminus e$.

Definition 4 (Edge Contraction (A/e)): The result of contracting edge $e \in \mathcal{E}$ from $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$, denoted by $A'' = A/e$, is $A'' = \text{minimal}(A')$, wherein $A' = A/e = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}', \beta)$ with $\mathcal{E}' = \mathcal{E} \setminus (e \cup \text{In}(Tl(e))) \cup_{e' \in \text{In}(Tl(e))} \{Tl(e'), \text{Hd}(e) \cup \text{Hd}(e')\}$.

It is important to recognize in the definitions above the minimality reduction operation, without it, it would be possible that the result of the deletion or contraction were non-minimal. Just as a graph minor can be created through a sequence of edge contractions and deletions, we will define an embedded network, or a network minor, to be the result of any sequence of the generalization of these operations to networks.

Definition 5 (Embedded Network): A network A' is said to be embedded in another network A , or equivalently is said to

be a minor of A , denoted as $A' \prec A$, if A' can be obtained by a sequence of operations of source deletions, edge deletions, and/or edge contractions on A . Furthermore, for two such networks $A' \prec A$, A is said to be an extension of A' , denoted $A \succ A'$.

The next collection of theorems track what happens to the rate region of a network as it undergoes a minor operation.

Theorem 3: Suppose a network $A'' = \text{minimal}(A')$ is a minimal form of a network A' created by deleting i , either a source $i = k$ defining $\rho_i = \omega_k$ or an edge $i = e$ defining $\rho_i = R_e$, from another network $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$, i.e., $A' = A \setminus i$. Then for every $l \in \{*, q, (s, q), o\}$

$$\mathcal{R}_l(A'') = \text{minimal}_{A' \rightarrow A''} \mathcal{R}_l(A'), \text{ with} \\ \mathcal{R}_l(A') = \text{Proj}_{(\omega, r) \setminus \rho_i} \{\mathbf{R} \in \mathcal{R}_l(A) \mid \rho_i = 0\}. \quad (20)$$

Proof: $\mathcal{R}_l(A') \subset \text{Proj}_{\omega \setminus H(Y_k), r} \{\mathbf{R} \in \mathcal{R}_l(A) \mid \rho_i = 0\}$ because any extreme point $(\omega', r') \in \mathcal{R}_l(A')$ is derived from an extreme point $\mathbf{h}' \in \Gamma_{N'}^l \cap \mathcal{L}_{A'}$, which can be extended to a point $\mathbf{h} \in \Gamma_N^l \cap \mathcal{L}_A$ via $\mathbf{h}_A = \mathbf{h}'_{A \setminus i}$ which yields $(\omega, r) = \text{Proj}_{(\omega, r)} \mathbf{h}$ with $\rho_i = 0$. Likewise, any point $(\omega, r) \in \mathcal{R}_l(A)$ with $\rho_i = 0$ can be derived from a $\mathbf{h} \in \Gamma_N^l \cap \mathcal{L}_A$, yielding \mathbf{h}' defined by $\mathbf{h}'_{A'} = \mathbf{h}_{A'}$, $\forall A' \subseteq \{1, \dots, N'\}$, with $\mathbf{h}' \in \Gamma_{N'}^l \cap \mathcal{L}_{A'}$ with $(\omega', r') = \text{Proj}_{\omega', r'} \mathbf{h}' = \text{Proj}_{(\omega, r) \setminus \rho_i} (\omega, r)$, which proves $\text{Proj}_{\omega \setminus H(Y_k), r} \{\mathbf{R} \in \mathcal{R}_l(A) \mid \rho_i = 0\} \subseteq \mathcal{R}_l(A')$. More details can be found in [26], [27]. ■

It is important to observe that, while their definition features a polyhedral projection, the inequality description for the rate region $\mathcal{R}_l(A')$ that is result of source or edge deletion (20), can be determined by simply substituting 0 in for the associated source or edge rate in the inequality description of the region $\mathcal{R}_l(A)$. As such, determining the resulting rate region is a low complexity operation.

Theorem 4: Suppose a network $A'' = \text{minimal}(A')$ is a minimal form of a network A' obtained by contracting e from another network $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$, i.e., $A' = A/e$. Then for

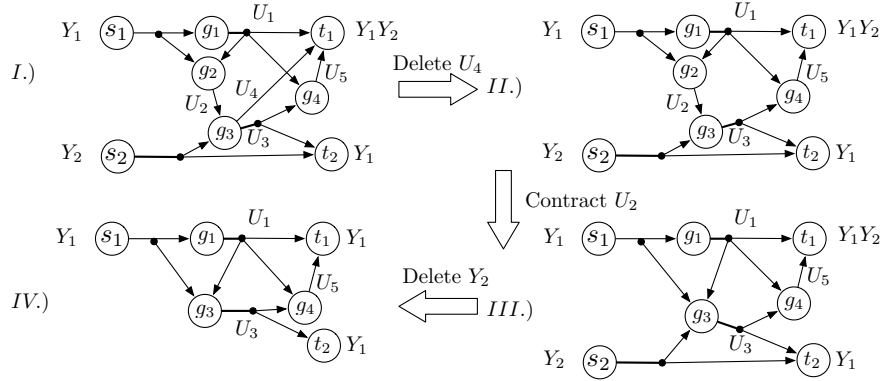


Figure 7: Using embedding operations to predict the insufficiency of scalar binary codes for a large network: since the large network I and intermediate stage networks II, III contain the small network IV as a minor, the insufficiency of scalar binary codes for network IV, which is easier than network I to see, predicts the same property for networks III, II and I.

Minimal (2,3) Networks:

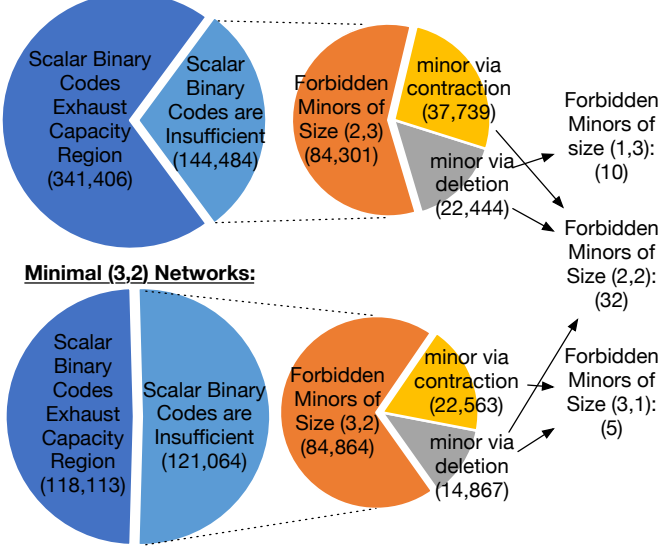


Figure 8: Relations between networks of different sizes that scalar binary codes do not suffice. The deletion operation considers both source and edge deletion, while the contraction operation only considers edge contraction.

$$l \in \{*, q, o\}$$

$$\mathcal{R}_l(A'') = \text{minimal}_{A' \rightarrow A''} \left(\text{Proj}_{\omega, r \setminus R_e} \mathcal{R}_l(A) \right), \quad (21)$$

$$\mathcal{R}_{s,q}(A'') \supseteq \text{minimal}_{A' \rightarrow A''} \left(\text{Proj}_{\omega, r \setminus R_e} \mathcal{R}_{s,q}(A) \right). \quad (22)$$

Proof: See Appendix B. ■

Note that the projection in (21) removes only one variable, R_e . As such, when $\mathcal{R}_l(A)$ is given, $\mathcal{R}_l(A')$ can be determined via (21) with a single step of Fourier-Motzkin elimination, and thus with substantially lower complexity than computing $\mathcal{R}_l(A')$ separately directly through (5)–(9).

A key implication of these theorems, summarized in the following corollary, is that sufficiency of a class of linear codes and/or tightness of the Shannon outer bound is preserved under the operation of taking minors.

Corollary 2: Consider two networks A, A' , with rate regions $\mathcal{R}_*(A), \mathcal{R}_*(A')$, such that $A' \prec A$. If \mathbb{F}_q vector (scalar) linear

codes suffice, or Shannon outer bound is tight for A , then same statements hold for A' . Equivalently, if \mathbb{F}_q vector (scalar) linear codes do not suffice, or Shannon outer bound is not tight for A' , then same statements hold for A . Equivalently, if $\mathcal{R}_l(A) = \mathcal{R}_*(A)$, then $\mathcal{R}_l(A') = \mathcal{R}_*(A')$, for some $l \in \{o, q, (s, q)\}$.

Proof: From Defn. 5, A' must be obtained by a series of source deletions, edge deletions, edge contractions, and minimality reductions. Theorems 3–4 indicate that sufficiency of linear codes, vector or scalar, and the tightness of Shannon outer bound are preserved for each single embedding operation. For vector case, if $\mathcal{R}_q(A) = \mathcal{R}_*(A)$, (20), (21) directly give $\mathcal{R}_q(A') = \mathcal{R}_*(A')$ for source deletion, edge contraction, and edge deletion, respectively. Similar arguments work for the tightness of the Shannon outer bound. For scalar code sufficiency, (20) indicate the same preservation of sufficiency of scalar codes for source and edge deletion, respectively. For edge contraction and assumption of if $\mathcal{R}_{s,q}(A) = \mathcal{R}_*(A)$, (21) and (22) indicate $\mathcal{R}_*(A') \subseteq \mathcal{R}_{s,q}(A')$. Together with the straightforward fact that $\mathcal{R}_{s,q}(A') \subseteq \mathcal{R}_*(A')$, we have $\mathcal{R}_{s,q}(A') = \mathcal{R}_*(A')$ holds for edge contraction as well. ■

As illustrated in Fig. 7, Corollary 2 enables us to explain characteristics of large networks, such as the lack of sufficiency of a class of linear codes, or tightness of the Shannon outer bound, as arising from smaller networks embedded within them.

In particular, the key hierarchical idea Corollary 2 inspires is that of a *forbidden network minor* for sufficiency of a class of codes or tightness of the Shannon outer bound. As taking a minor of a particular network coding problem preserves these properties, we can properly study those networks that do not have these properties by studying the smallest problems exhibiting them. These can be thought of as network minors that a larger network is forbidden to have if the specified property is desired, because the negation has now enabled the implications to work in the opposite direction: if a larger network contains this small network, it too must lack the specified property.

Definition 6 (Forbidden Network Minor): A *forbidden network minor* for a specified desired network characteristic, such as the sufficiency of a class of linear codes, or tightness of the

Shannon outer bound, is a network lacking this property, for whom all minors have the property.

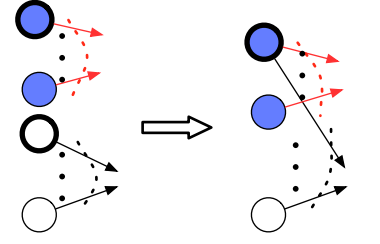
This idea has two immediate uses with respect to learning from large databases of complicated small network coding problems whose rate regions have been proven by a computer, while a third will be discussed in §VIII. The first use is to organize the results in the database via the implications set up by the minor relationships. As an example, Fig. 8 studies the property of sufficiency of scalar binary codes within the database described in §V. The organizational principle provided by minor relationships is very powerful: 96613 network coding problems of size (2, 3) and (3, 2) for whom binary scalar codes not not exhaust the capacity region are directly implied from only 47 smaller networks with this property. The second use is to determine properties of networks at scale from the information learned from the database using the implications that the minor relationship provides. For instance, the 47 tiny forbidden minor networks, along with 169165 more forbidden minor networks of size (3, 2) and (2, 3) summarized in Fig. 8, can be used to decisively declare the insufficiency of binary codes to exhaust capacity regions of an arbitrary number of networks of arbitrarily large scale and size.

VII. NETWORK COMBINATION OPERATIONS

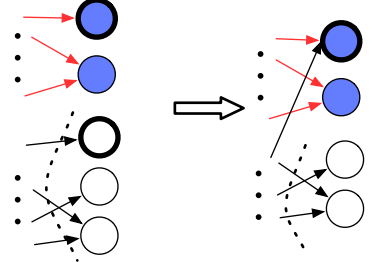
A drawback of using embedding operations from §VI alone to learn from a database of small network coding rate regions is that, aside from an organization principle, only negative results – e.g. insufficiency of a class of linear codes, or looseness of the Shannon outer bound – can be inferred about large networks via corollary 2. To address this, in this section, we develop operators which can work in the opposite direction. In particular, we view the database of small networks as building blocks, and we construct rules for constructing larger networks via combinations of smaller ones in a manner enabling the capacity region of the large network to be inferred through some low complexity calculations with capacity regions of the small ones. While several previous works, e.g. [48], have presented other methods for pasting networks together with different aims, the operations here will be restricted to be ones that enable the capacity region of the resulting network to be easily determined.

The following definitions provide ways for constructing a network $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}, \beta)$ as a combination of two *disjoint* networks $A_i = (\mathcal{S}_i, \mathcal{G}_i, \mathcal{T}_i, \mathcal{E}_i, \beta_i)$, $i \in \{1, 2\}$, meaning $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$, $\mathcal{G}_1 \cap \mathcal{G}_2 = \emptyset$, $\mathcal{T}_1 \cap \mathcal{T}_2 = \emptyset$, $\mathcal{E}_1 \cap \mathcal{E}_2 = \emptyset$, and $\beta_1(t_1) \cap \beta_2(t_2) = \emptyset, \forall t_1 \in \mathcal{T}_1, t_2 \in \mathcal{T}_2$. The combinations we will define will merge network elements, i.e., sources, intermediate nodes, sink nodes, edges, etc, and are depicted in Fig. 9. Since each merge will combine one or several pairs of elements, with each pair containing one element from A_1 and the other from A_2 , each merge definition will involve a bijection π indicating which element from the appropriate set of A_2 is paired with its argument in A_1 .

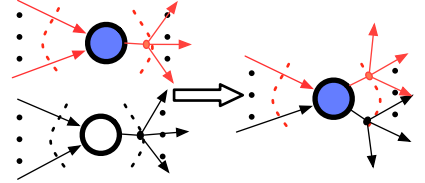
Definition 7 (Source Merge ($A_1.\hat{\mathcal{S}} = A_2.\pi(\hat{\mathcal{S}})$) – Fig. 9a): Merging the sources $\hat{\mathcal{S}} \subseteq \mathcal{S}_1$ from network A_1 with the sources $\pi(\hat{\mathcal{S}}) \subseteq \mathcal{S}_2$ from a disjoint network A_2 , will produce a network A with



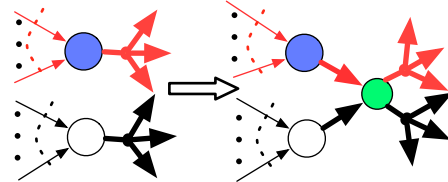
(a) Sources merge: the merged source will serve for the new larger network.



(b) Sinks merge: input and output of the sinks are unioned, respectively.



(c) Intermediate nodes merge: input and output of the nodes are unioned, respectively.



(d) Edges merge: one extra node and four associated edges are added to replace the two edges.

Figure 9: Combination operations on two smaller networks to form a larger network. Thickly lined nodes (edges) are merged.

i) merged sources $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \setminus \pi(\hat{\mathcal{S}})$,

ii) $\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2$,

iii) $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$,

iv) $\mathcal{E} = (\mathcal{E}_1 \cup \mathcal{E}_2 \setminus \mathcal{A}) \cup \mathcal{B}$,

where $\mathcal{A} = \{e \in \mathcal{E}_1 \cup \mathcal{E}_2 \mid \text{tl}(e) \in \hat{\mathcal{S}} \cup \pi(\hat{\mathcal{S}})\}$ includes the edges connected with the sources involved in the merge, $\mathcal{B} = \{(s, \mathcal{F}_1 \cup \mathcal{F}_2) \mid s \in \hat{\mathcal{S}}, (s, \mathcal{F}_1) \in \mathcal{E}_1, (\pi(s), \mathcal{F}_2) \in \mathcal{E}_2\}$ includes the new edges connected with the merged sources, and

v) updated sink demands

$$\beta(t) = \begin{cases} \beta_1(t) & t \in \mathcal{T}_1 \\ (\beta_2(t) \setminus \pi(\hat{\mathcal{S}})) \cup \pi^{-1}(\pi(\hat{\mathcal{S}}) \cap \beta_2(t)) & t \in \mathcal{T}_2 \end{cases}.$$

Definition 8 (Sink Merge ($A_1.\hat{\mathcal{T}} + A_2.\pi(\hat{\mathcal{T}})$) – Fig. 9b.): Merging the sinks $\hat{\mathcal{T}} \subseteq \mathcal{T}_1$ from network A_1 with the sinks $\pi(\hat{\mathcal{T}}) \subseteq \mathcal{T}_2$ from the disjoint network A_2 will produce a network A with

- i) $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$; $\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2$,
- ii) $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2 \setminus \pi(\hat{\mathcal{T}})$,
- iii) $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2 \cup \mathcal{A} \setminus \mathcal{B}$,
 where $\mathcal{A} = \{(g_2, \mathcal{F}_1 \cup \mathcal{F}_2) | g_2 \in \mathcal{G}_2, \mathcal{F}_1 \subseteq \hat{\mathcal{T}}, \mathcal{F}_2 \subseteq \mathcal{T}_2, (g_2, \pi(\mathcal{F}_1) \cup \mathcal{F}_2) \in \mathcal{E}_2\}$ updates the head nodes of edges in \mathcal{A}_2 with new merged sinks, $\mathcal{B} = \{(g_2, \mathcal{F}_2) \in \mathcal{E}_2 | \mathcal{F}_2 \cap \pi(\hat{\mathcal{T}}) \neq \emptyset\}$ includes the edges connected to sinks in $\pi(\hat{\mathcal{T}})$, and
- v) updated sink demands

$$\beta(t) = \begin{cases} \beta_i(t) & t \in \mathcal{T}_i \setminus \hat{\mathcal{T}}, i \in \{1, 2\} \\ \beta_1(t) \cup \beta_2(\pi(t)) & t \in \hat{\mathcal{T}} \end{cases}. \quad (23)$$

Definition 9 (Intermediate Node Merge $(A_1.g + A_2.\pi(g))$ – Fig. 9c): Merging the intermediate node $g \in \mathcal{G}_1$ from network A_1 with the intermediate node $\pi(g) \in \mathcal{G}_2$ from the disjoint network A_2 will produce a network A with i) $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$, ii) $\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2 \setminus \pi(g)$, iii) $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$, iv) $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2 \cup \mathcal{A} \cup \mathcal{B} \setminus \mathcal{C} \setminus \mathcal{D}$, where $\mathcal{A} = \{(g_2, \mathcal{F}_2 \setminus \pi(g) \cup g) | g_2 \in \mathcal{G}_2, (g_2, \mathcal{F}_2 \cup \pi(g)) \in \mathcal{E}_2\}$ updates the head nodes of edges in \mathcal{A}_2 that have $\pi(g)$ as head node, $\mathcal{B} = \{(g, \mathcal{F}_2) | (\pi(g), \mathcal{F}_2) \in \mathcal{E}_2\}$ updates the tail node of edges in \mathcal{A}_2 that have $\pi(g)$ as tail node, $\mathcal{C} = \{e \in \mathcal{E}_2 | \text{TI}(e) = \pi(g)\}$ includes the edges in \mathcal{A}_2 that have $\pi(g)$ as tail node, $\mathcal{D} = \{e \in \mathcal{E}_2 | \pi(g) \in \text{Hd}(e)\}$ includes the edges in \mathcal{A}_2 that have $\pi(g)$ as head node; and v) updated sink demands

$$\beta(t) = \begin{cases} \beta_1(t) & t \in \mathcal{T}_1 \\ \beta_2(t) & t \in \mathcal{T}_2 \end{cases} \quad (24)$$

Definition 10 (Edge Merge $(A_1.e + A_2.\pi(e))$ – Fig. 9d): Merging edge $e \in \mathcal{E}_1$ from network A_1 with edge $\pi(e) \in \mathcal{E}_2$ from disjoint network A_2 will produce a network A with i) $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$, ii) $\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup g_0$, where $g_0 \notin \mathcal{G}_1, g_0 \notin \mathcal{G}_2$, iii) $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$, iv) $\mathcal{E} = (\mathcal{E}_1 \setminus e) \cup (\mathcal{E}_2 \setminus \pi(e)) \cup \{(\text{TI}(e), g_0), (\text{TI}(\pi(e)), g_0), (g_0, \text{Hd}(e)), (g_0, \text{Hd}(\pi(e))) \}$; and v) updated sink demands given by (24).

The following theorems determine the manner in which the rate region of the result of a combination operation between networks can be determined from the rate regions of the arguments of the operation.

Theorem 5: Suppose a network A is obtained by merging the set of source nodes $\hat{\mathcal{S}}$ with $\pi(\hat{\mathcal{S}})$, i.e., $A_1.\hat{\mathcal{S}} = A_2.\pi(\hat{\mathcal{S}})$. Then $\forall l \in \{*, q, (s, q), o\}$

$$\mathcal{R}_l(A) = \text{Proj}_{\omega, r}((\mathcal{R}_l(A_1) \times \mathcal{R}_l(A_2)) \cap \mathcal{L}_0). \quad (25)$$

$$\text{with } \mathcal{L}_0 = \left\{ H(Y_s) = H(Y_{\pi(s)}), \forall s \in \hat{\mathcal{S}} \right\}.$$

Proof: The essence of the proof is that, owing to the disjoint nature of the two networks A_1 and A_2 , a putative vector (ω, r) is in $\mathcal{R}_l(A)$ if and only if its components corresponding to A_1 are in $\mathcal{R}_l(A_1)$ and its components corresponding to A_2 are in $\mathcal{R}_l(A_2)$. A detailed proof is available in [26], [27]. ■

It is important to note that combining rate regions under merging of sources is a low complexity operation as is indicated by the following remark.

Remark 1: The inequality description of the convex cone $\text{Proj}_{\omega, r}((\mathcal{R}_l(A_1) \times \mathcal{R}_l(A_2)) \cap \mathcal{L}_0)$ can be created by concatenating the inequality descriptions for $\mathcal{R}_l(A_1)$ and $\mathcal{R}_l(A_2)$, then replacing the variable $H(Y_{\pi(s)})$ with the variable $H(Y_s)$ for each $s \in \hat{\mathcal{S}}$. As such (25) is a low complexity operation.

Theorem 6: Suppose a network A is obtained by merging a set of sink nodes $\hat{\mathcal{T}}$ with $\pi(\hat{\mathcal{T}})$, i.e., $(A_1.\hat{\mathcal{T}} + A_2.\pi(\hat{\mathcal{T}}))$. Then

$$\mathcal{R}_l(A) = \mathcal{R}_l(A_1) \times \mathcal{R}_l(A_2), \quad l \in \{*, q, (s, q), o\} \quad (26)$$

with the index on the dimensions mapping from $\{e \in \mathcal{E}_2 | \text{Hd}(e) \in \pi(\hat{\mathcal{T}})\}$ to $\{e \in \mathcal{E} | \text{Hd}(e) \in \hat{\mathcal{T}}, \text{TI}(e) \in \mathcal{G}_2\}$.

Proof: As no path from sources from A_2 meets a path from sources A_1 until a sink node, and the sources are independent, the random variables in A can be partitioned into two disjoint sets $\mathcal{N} = \mathcal{N}_1 \cup \mathcal{N}_2$, those derived from A_1, \mathcal{N}_1 , and those derived from A_2, \mathcal{N}_2 . The essence of the argument is that the sources and messages arriving at the sink that are from the part of the network from A_1 are independent of the sources and messages arriving from A_2 , with any $\mathbf{h} \in \Gamma_N^l$ having $h_{\mathcal{A}} = h_{\mathcal{A} \cap \mathcal{N}_1} + h_{\mathcal{A} \cap \mathcal{N}_2}$ for all $\mathcal{A} \subseteq \mathcal{N}$. As such, the decoding constraints for A can be rewritten as the decoding constraints for A_1 and A_2 separately, and the remaining constraints in \mathcal{L}_A separate into the constraints for A_1 and A_2 as well. Defining \mathbf{h}_i having elements $h_{i, \mathcal{A}} = h_{\mathcal{A}}, \mathcal{A} \subseteq \mathcal{N}_i$ for all $i \in \{1, 2\}$, we observe then that $\mathbf{h}_i \in \Gamma_N^l \cap \mathcal{L}_{A_i}$, and $h_{\mathcal{A}} = h_{1, \mathcal{A} \cap \mathcal{N}_1} + h_{2, \mathcal{A} \cap \mathcal{N}_2}, \mathcal{A} \subseteq \mathcal{N}$, with \mathbf{h}_i projecting to a point in $\mathcal{R}(A_i)$, and proving (26). A detailed proof is available in [26], [27]. ■

While entire sets of sources and sinks can be merged at once and the rate region of the result can be rapidly determined through Thm. 5 and Thm. 6, a key restriction in the corresponding results for intermediate node and edge merge operations is that only one node or edge between the disjoint networks can be merged. This restriction is in place because without it, the rate region of the result can no longer be inferred from the rate region of the component networks.

Theorem 7: Suppose a network A is obtained by merging the intermediate node g with $\pi(g)$, i.e., $A_1.g + A_2.\pi(g)$. Then

$$\mathcal{R}_l(A) = \mathcal{R}_l(A_1) \times \mathcal{R}_l(A_2), \quad l \in \{*, q, (s, q), o\} \quad (27)$$

with dimensions/ indices mapping from $\{e \in \mathcal{E}_2 | \text{Hd}(e) = \pi(g)\}$ to $\{e \in \mathcal{E} | \text{Hd}(e) = g, \text{TI}(e) \in \mathcal{G}_2\}$ and from $\{e \in \mathcal{E}_2 | \text{TI}(e) = \pi(g)\}$ to $\{e \in \mathcal{E} | \text{TI}(e) = g, \text{Hd}(e) \in \mathcal{G}_2\}$.

Proof: The essence of the argument for $l \in \{*, (s, q), q\}$ is that paths in A from sources in A_1 to sinks in A_1 can exclusively meet path from sources from A_2 to sinks from A_2 at g . As such outgoing edges from g can be partitioned into those going to nodes from A_i and those going to nodes from A_{3-i} , and selecting an erroneous (e.g. constant) value of the incoming sources/messages in A_{3-i} for those messages leaving for A_i must not alter correct decoding, as decoding must be done at those sinks regardless of those values, for both $i \in \{1, 2\}$. For $\mathcal{R}_o(A)$, partitioning the message and source variables \mathcal{N} in A into those parts \mathcal{N}_i associated with $A_i, i \in \{1, 2\}$, observe that if $\mathbf{h} \in \Gamma_N^l \cap \mathcal{L}_A$, then so is \mathbf{h}' defined through

$$h'_{\mathcal{A}} = (h_{(\mathcal{A} \cap \mathcal{N}_1 \cup \mathcal{S}_2)} - h_{\mathcal{S}_2}) + (h_{(\mathcal{A} \cap \mathcal{N}_2 \cup \mathcal{S}_1)} - h_{\mathcal{S}_1}), \quad (28)$$

which gives the same source rates as \mathbf{h} and dominating edge rates, while $h_{i, \mathcal{A}} = h_{(\mathcal{A} \cap \mathcal{N}_i \cup \mathcal{S}_{3-i})} - h_{\mathcal{S}_{3-i}}, \mathcal{A} \subseteq \mathcal{N}_i$ is in $\Gamma_N^o \cap \mathcal{L}_{A_i}$, for $i \in \{1, 2\}$, proving $\mathcal{R}_o(A) \subseteq \mathcal{R}_o(A_1) \times \mathcal{R}_o(A_2)$. More details are available in [26], [27]. ■

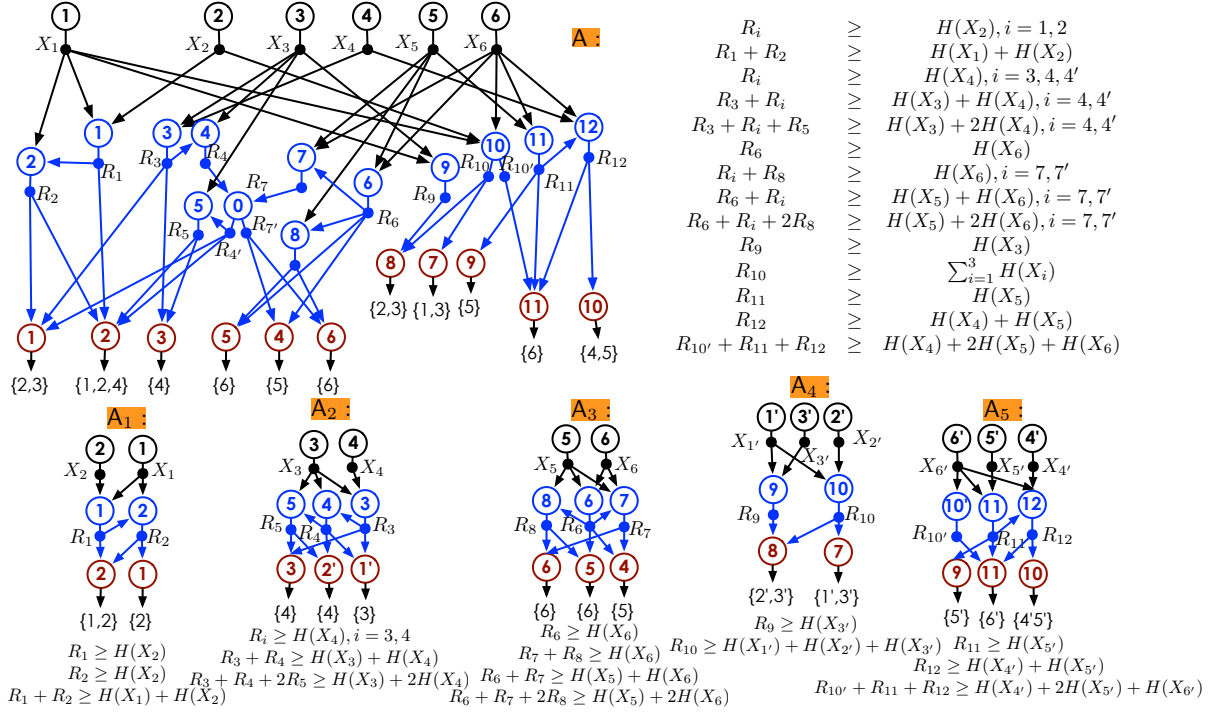


Figure 10: A large network and its rate region created with the operations in this paper from the 5 networks below it.

Theorem 8: Suppose a network A is obtained by merging e and $\pi(e)$, i.e., $A_1.e + A_2.\pi(e)$. Then $l \in \{*, q, (s, q), o\}$

$$\mathcal{R}_l(A) = \text{Proj}_{\omega, r}((\mathcal{R}_l(A_1) \times \mathcal{R}_l(A_2)) \cap \mathcal{L}'_0) \quad (29)$$

wherein

$$\mathcal{L}'_0 = \{R_{(Tl(j), g_0)} \geq R_j, R_{(g_0, Hd(j))} \geq R_j, j \in \{e, \pi(e)\}\},$$

and only the dimensions R_e and $R_{\pi(e)}$ are being eliminated in the projection $\text{Proj}_{\omega, r}$.

Proof: Edge merge can be built from intermediate node merge as follows. First replace e in A_1 with two edges $(Tl(e), g)$ and $(g, Hd(e))$ for a new node g in A_1 to get A'_1 , and replace $\pi(e)$ with two edges $(Tl(\pi(e)), g')$ and $(g', Hd(\pi(e)))$ for a new node g' . Then merging g and g' give the theorem. More details are available in [26], [27]. ■

It is important to note that propagating rate regions through and edge merge is a much lower complexity operation than calculating the rate region for the large network from scratch.

Remark 2: While a projection operator is featured in (29), unlike the projection in (2) and (4) which must eliminate a number of dimensions that is exponential in the size of the network, only two dimensions are being removed in (29). As such, using the computations (29) to derive the capacity region of the resulting network A from those of A_1 and A_2 , has drastically lower complexity than deriving it from scratch via (2) and (4).

A key implication of these theorems tracking rate regions through combination operations, summarized in the following corollary, is that while passing from smaller networks to larger ones, key characteristics are preserved.

Corollary 3: Let network A be a combination of networks A_1, A_2 via one of the combination operations. If \mathbb{F}_q vector (scalar) linear codes suffice or the Shannon outer bound is tight for both A_1, A_2 , then the same will be true for A and any minor of it $A' \prec A$. Equivalently, if $\mathcal{R}_l(A_i) = \mathcal{R}_*(A_i), i \in \{1, 2\}$ for some $l \in \{o, q, (s, q)\}$ then also $\mathcal{R}_l(A) = \mathcal{R}_*(A)$ and $\mathcal{R}_l(A') = \mathcal{R}_*(A')$.

An important application of combination operators is that they enable us to rapidly infer the rate regions of a certain class of arbitrarily large networks through a sequence of low complexity operations combining small networks. This enables a database of relatively small network coding problems to reach applications in networks of much larger size. Example 2 provides an explicit example of this – the network discussed therein is already far too large to enable its rate regions to be directly calculated through verbatim calculation of the projection displayed in 5–9, yet the combination operator have enabled its capacity region to be determined as a combination of answers from the database.

Example 2: A (6, 15) network instance A can be obtained by combining five smaller networks A_1, \dots, A_5 , of which the representations are shown in Fig. 10. The combination process is I) $A_{12} = A_1.\{t_1, t_2\} + A_2.\{t'_1, t'_2\}$; II) $A_{123} = A_{12}.e_4 + A_3.e_7$ with extra node g_0 and edges $e_{4'}, e_{7'}$; III) $A_{45} = A_4.g_{10} + A_5.g_{10'}$; IV) $A = A_{123}.\{X_1, \dots, X_6\} = A_{45}.\{X_{1'}, \dots, X_{6'}\}$. From the software calculations and analysis [58], [59], one obtains the rate regions below the 5 small networks. According to the theorems in §VII, the rate region $\mathcal{R}_*(A)$ for A obtained from $\mathcal{R}_*(A_1), \dots, \mathcal{R}_*(A_5)$, is depicted next to it. Additionally, since calculations showed binary codes and the Shannon outer

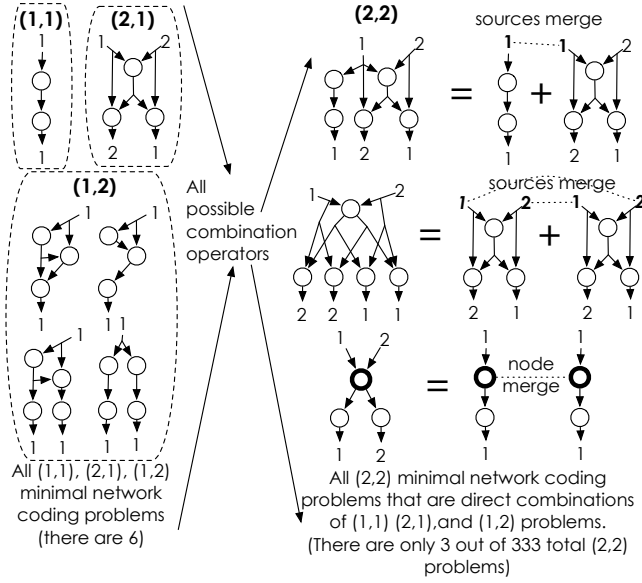


Figure 11: There are a total of 3 minimal (2,2) network coding problems directly resulting from combinations of the 6 small network coding problems with sizes (1,1), (1,2), and (2,1). However, as shown in Fig. 12, by utilizing both combinations and embeddings operators, far more (2,2) cases can be reached by iteratively combining and embedding the pool of networks starting from these 6 (1,1), (1,2), and (2,1) networks via Algorithm 2.

bound suffice for A_i , $i \in \{1, \dots, 5\}$, Corollary 3 dictates the same for network A .

VIII. COMBINATION AND EMBEDDING OPERATORS ACT TOGETHER

The combination operators presented in §VII provide a series of rules for combining solutions for rate regions of small networks to get rate regions of larger networks. In principle these combinations operators alone enable large databases of rate regions of small minimal networks, listed with the methods in §IV and solved with ITAP and ITCP, exemplified by the one described in §V, to be used as building blocks which can be put together to solve far larger networks. However, the coverage of all possible network coding problems that these combinations operators alone can solve is somewhat small, owing to the somewhat restricted ways in which they enable networks to be pasted together. For instance, Fig. 11 shows that there are only three possible (2,2) networks that can be created by acting with the combination operators combining all pairs between the six smallest minimal networks, while we know from Table III that there 333 minimal (2,2) networks.

On the other hand, the results provided in §VI also enable the rate region to be tracked through an embedding operation that shrinks a network. Thus, the rate region of a result of a sequence of both combination and embedding operations can be inferred with low complexity from the rate regions supplied to the original combinations operator, simply by performing the low complexity calculations prescribed by the appropriate theorem (Thm.s 3–8) for each operation in the sequence. As each of these operations themselves is of low complexity, this

provides a method for solving far larger networks than can be reached by direct solution of (5)–(9). In this section, we provide some results showing that many more networks can be reached by using both combinations and embedding operations together than can be reached by combinations alone.

Indeed, Fig. 12 illustrates the simplest and smallest case of this fact, three examples of (2,2) networks whose rate regions can be created by a sequence of combination and embedding networks among the six smallest networks from Fig. 11, but all three of which are not among the (2,2) networks in Fig. 11 that could be reached from combinations operations alone. Thus, starting from a seed database of the rate regions of even just the six smallest networks, the solutions for these three new (2,2) networks can also be determined with far lower complexity than calculating their rate regions from scratch using (5)–(9).

From this fact that combinations and embeddings together can reach a larger collection of problems, the question arises as how to formalize assessing how many networks this process of a sequence of combinations and embedding operations can reach. Clearly, as the same network can be combined with itself and infinite number of times, an infinite number of networks can be reached, however, of greater interest is the fraction of networks of a particular size that can be reached by such a process, starting only with the networks of smaller sizes. Even here, in order for the process to be answered in a finite amount of time, it will be necessary to cap the size of the network encountered among the combinations and embeddings. This yields the algorithm of *partial operator closure*, depicted in Alg. 2, which finds all minimal networks that can be reached by a sequence of combinations and embedding operations among a seed list of initial networks and their rate regions, only considering combination operations whose largest possible results would not pass a certain threshold size, the cap.

To illustrate the sheer power of combinations and embedding operators alone to generate rate regions, we have listed the result of running this partial network closure operation on the six smallest networks depicted in Fig. 11 in Table V. Several important facts can be learned from Table V. First of all, it is evident by comparing the three columns at the right that the number of networks, even of a fixed small size, that are reachable with combination and embedding operators increases with the cap size of the largest network encountered in the process, with, e.g., the number of (2,3) networks whose rate regions can be determined by combinations among the 6 smallest minimal networks increasing from 33 at a cap size of (3,3) to (155) at a cap size of (4,4). Second of all, by comparing the three columns at the left with the three on the right it is evident that there is a huge benefit from using both combinations and embedding operators, in the sense that many more networks can be reached. Finally, we observe that calculating network coding rate regions through combinations and embedding operations using a seed list that is a database of all small networks up to a given size, such as the one in §V, will handle an incredible number of networks, as the number in the bottom right corner indicates that using even only the six smallest networks can reach 11635 networks with a small

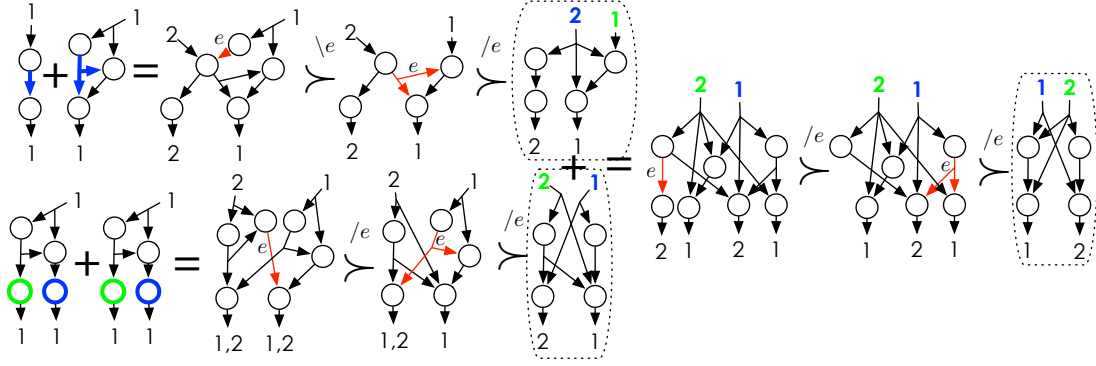


Figure 12: The path of operations on a seed list of small networks – the $(1,1)$, $(1,2)$, $(2,1)$ minimal networks in Fig. 11 – to get three $(2,2)$ networks, outlined with dotted lines, that cannot be directly obtained by simple combination. The size limits on networks involved in the operation process is $K \leq 3, L \leq 4$.

cap size.

IX. CONCLUSIONS AND FUTURE WORK

Recognizing the recent development of algorithms and software [60], [61] that can generate and prove the capacity regions of small networks under network coding, this article set about developing a theory that can best exploit this newfound capability to learn about capacity regions of larger networks. First, in §III, a series of 14 minimality conditions were listed that removed inessential components from a network coding problem, along with a method of determining the rate region of non-minimal network from the minimal one. Next, in §IV, observing that multiple replicas of the same essential minimal network coding problem exist under various methods of labeling the network, a method for directly listing only one representative from each equivalence class of minimal networks under this relabelling was provided. This method, together with the rate region proving software, then enabled the rate regions for all 744,119 equivalence classes of minimal networks with the sum of sources and edges less than or equal to 5 to be determined in §V. This database of rate regions showed that for all of these problems, linear codes suffice and the Shannon outer bound is tight.

However, the desire to organize this database of small networks such as these, and to learn from them characteristics of networks at scale inspired us to formalize the notion of embedded networks in §VI. Embedding operators were developed that recognized small network coding problems included within larger ones in such a manner that the rate region of the smaller problem could be directly inferred from the larger one, and the sufficiency of certain class of linear codes or Shannon type inequalities was inherited by the smaller network from the larger one. This, in turn, enabled us to predict and explain characteristics of arbitrarily large networks through the language forbidden network minors – small problems that large network could not contain if a certain property such as sufficiency of a class of codes or tightness of the Shannon outer bound were desired. However, observing that these were only negative results about networks at scale, in §VII we next defined a series of combinations operators, which showed how to paste together small networks

Input: Seed list of networks $seedList$, size limits on number of sources and edges
Output: All network instances generated by combination and/or embedding operations on the seed list
Initialization: network list for previous round
 $prevList = \emptyset$, new networks from previous round
 $prevAdd = seedList$, current list of networks
 $curList = \emptyset$, new networks generated in current round
 $curAdd = \emptyset$;
while $size(prevAdd) > 0$ **do**
 for every pair
 $\mathcal{I} \times \mathcal{J} \in prevAdd \times prevAdd \cup prevAdd \times curList$
 do
 if prediction of network size after merge does not exceed size limits **then**
 consider source, sink, node, edge merge on \mathcal{I}, \mathcal{J} ;
 convert the new network to its canonical form $newNet$;
 if $newNet \notin curList$ **then**
 $curAdd = curAdd \cup newNet$;
 end
 end
 end
 for every $\mathcal{I} \in prevAdd$ **do**
 consider source deletion, edge deletion and edge contraction on \mathcal{I} ;
 convert the new network to its canonical form $newNet$;
 if $newNet \notin curList$ **then**
 $curAdd = curAdd \cup newNet$;
 end
 end
 $prevAdd = curAdd$;
 $prevList = curList$;
 $curList = curList \cup curAdd$;
end

Algorithm 2: Generate all networks from a seed list of small networks using combination and embedding operations.

Table V: The number of new canonical minimal network coding problems that can be generated from the 6 smallest canonical minimal network coding problems (the single (1, 1) network, the single (2, 1) network, and the four (1, 2) networks), by using combination operators (left), and both combination and embedding operators (right), in a partial closure operation where the largest network involved in a chain of operations never exceeds the “cap” (different columns).

Size/Cap	combination operators only			embedding and combinations		
	(3,3)	(3,4)	(4,4)	(3,3)	(3,4)	(4,4)
(1,3)	4	4	4	4	4	4
(1,4)	0	10	10	0	10	10
(2,2)	3	3	3	8	15	16
(2,3)	16	16	16	33	131	155
(2,4)	0	97	101	0	516	648
(3,2)	2	2	2	4	10	11
(3,3)	24	24	24	42	353	833
(3,4)	0	135	135	0	2361	5481
(4,2)	0	0	3	0	0	3
(4,3)	0	0	17	0	0	44
(4,4)	0	0	253	0	0	4430
all	49	291	568	91	3400	11635

into larger ones in such a manner that the rate region of the larger network could be directly inferred from the rate regions of the smaller networks. We then showed in §VIII, that both combinations and embedding operations could be used together to solve new networks that combinations could not solve alone, defining a notion of partial network operator closure. This enables a database of solved network coding problems to be used to generate, through combinations and embeddings, the rate regions of an arbitrarily large number of arbitrarily large networks.

These operations open a door to many new avenues of network coding research. Some of the pressing future problems for investigation include: I) assessing the coverage of the operators in the space of all problems; II) if necessary, the creation of more powerful combination operations, such as node and edge merge, source and sink merge, etc; III) a notion of forbidden minors which can harness both combination and embedding operators.

REFERENCES

- [1] C. Li, S. Weber, and J. Walsh, “Network embedding operations preserving the insufficiency of linear network codes,” in *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2014, Oct 2014.
- [2] C. Li, S. Weber, and J. M. Walsh, “Computer aided proof for rate regions of independent distributed source coding problems,” in *IEEE International Symposium on Network Coding (NetCod)*, Jun 2015.
- [3] C. Li, S. Weber, and J. Walsh, “Network combination operations preserving the sufficiency of linear network codes,” in *IEEE Information Theory Workshop (ITW)*, 2015, submitted.
- [4] T. Chan and A. Grant, “Dualities between entropy functions and network codes,” *IEEE Transactions on Information Theory*, vol. 54, no. 10, pp. 4470–4487, October 2008.
- [5] T. Chan and A. Grant, “Entropy Vectors and Network Codes,” in *IEEE International Symposium on Information Theory*, Jun. 2007.
- [6] T. Chan and A. Grant, “Dualities between entropy functions and network codes,” in *Fourth Workshop on Network Coding, Theory and Applications (NetCod)*, January 2008.
- [7] —, “Mission impossible: computing the network coding capacity region,” in *IEEE International Symposium on Information Theory (ISIT)*, July 2008, pp. 320–324.
- [8] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, “Network information flow,” *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, Jul 2000.
- [9] R. Dougherty, C. Freiling, and K. Zeger, “Networks, matroids, and non-shannon information inequalities,” *IEEE Transactions on Information Theory*, vol. 53, no. 6, pp. 1949–1969, 2007.
- [10] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, Sept 2010.
- [11] C. Tian, “Characterizing the rate region of the (4,3,3) exact-repair regenerating codes,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 967–975, May 2014.
- [12] J. Walsh, S. Weber, and C. Maina, “Optimal rate–delay tradeoffs and delay mitigating codes for multipath routed and network coded networks,” *Information Theory, IEEE Transactions on*, vol. 55, no. 12, pp. 5491–5510, Dec 2009.
- [13] S. Weber, C. Li, and J. M. Walsh, “Rate region for a class of delay mitigating codes and p2p networks,” *46th Annual Conference on Information Sciences and Systems*, March 2012.
- [14] D. Leong, A. Qureshi, and T. Ho, “On coding for real-time streaming under packet erasures,” in *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, July 2013, pp. 1012–1016.
- [15] X. Yan, R. Yeung, and Z. Zhang, “An implicit characterization of the achievable rate region for acyclic multisource multisink network coding,” *IEEE Transactions on Information Theory*, vol. 58, no. 9, pp. 5625–5639, Sept 2012.
- [16] Z. Zhang and R. Yeung, “On characterization of entropy function via information inequalities,” *Information Theory, IEEE Transactions on*, vol. 44, no. 4, pp. 1440–1452, Jul 1998.
- [17] R. W. Yeung, *Information Theory and Network Coding*. New York, US: Springer, 2008.
- [18] Jayant Apte and John MacLaren Walsh, “Explicit Polyhedral Bounds on Network Coding Rate Regions via Entropy Function Region: Algorithms, Symmetry, and Computation,” *IEEE Trans. Inf. Theory*, 2016, Submitted July 22, 2016. [Online]. Available: <http://arxiv.org/abs/1607.06833>
- [19] —, “Constrained Linear Representability of Polymatroids and Algorithms for Computing Achievability Proofs in Network Coding,” *IEEE Trans. Inf. Theory*, Submitted May 15, 2016. [Online]. Available: <http://arxiv.org/abs/1605.04598>
- [20] C. Li, S. Weber, and J. M. Walsh, “Multilevel diversity coding systems: Rate regions, codes, computation, forbidden minors,” *IEEE Transactions on Information Theory*, vol. 63, no. 1, pp. 230–251, 2017.
- [21] C. Li, J. Apte, J. M. Walsh, and S. Weber, “A new computational approach for determining rate regions and optimal codes for coded networks,” in *IEEE International Symposium on Network Coding (NetCod)*, Jun 2013, pp. 1–6.
- [22] C. Li, J. M. Walsh, and S. Weber, “Computational approaches for determining rate regions and codes using entropic vector bounds,” in *50th Annual Allerton Conference on Communication, Control and Computing*, Oct 2012, pp. 1–9.
- [23] Jayant Apte and John MacLaren Walsh, “Information Theoretic Converse Prover.” [Online]. Available: <https://github.com/jayant91089/itcp>
- [24] —, “Information Theoretic Achievability Prover.” [Online]. Available: <https://github.com/jayant91089/itap>
- [25] G. Gallo, G. Longo, S. Pallottino, and S. Nguyen, “Directed hypergraphs and applications,” *Discrete applied mathematics*, vol. 42, no. 2, pp. 177–201, 1993.
- [26] Congduan Li, “On Multi-source Multi-Sink Hyperedge Networks: Enumeration, Rate Region Computation, and Hierarchy,” Ph.D. dissertation, Drexel University, Philadelphia, PA, 2015. [Online]. Available: http://www.ece.drexel.edu/walsh/CongduanLi_PhD.pdf
- [27] C. Li, S. Weber, and J. M. Walsh, “On multi-source networks: Enumeration, rate region computation, and hierarchy,” *CoRR*, vol. abs/1507.05728, 2015. [Online]. Available: <http://arxiv.org/abs/1507.05728>
- [28] Raymond W. Yeung, “A Framework for Linear Information Inequalities,” *IEEE Trans. on Information Theory*, vol. 43, no. 6, Nov. 1997.
- [29] Zhen Zhang and Raymond W. Yeung, “On Characterization of Entropy Function via Information Inequalities,” *IEEE Trans. on Information Theory*, vol. 44, no. 4, Jul. 1998.
- [30] R. Dougherty, C. Freiling, and K. Zeger, “Linear rank inequalities on five or more variables,” *arXiv cs.IT/0910.0284v3*, 2009.

- [31] R. Kinser, “New inequalities for subspace arrangements,” *Journal of Combinatorial Theory Series A*, vol. 118, no. 1, pp. 152–161, January 2011.
- [32] J. Apte and J. Walsh, “Symmetry in network coding,” in *2015 IEEE International Symposium on Information Theory (ISIT)*, June 2015.
- [33] J. Apte and J. M. Walsh, “Exploiting symmetry in computing polyhedral bounds on network coding rate regions,” in *IEEE International Symposium on Network Coding (NetCod)*, Jun 2015.
- [34] A. Betten, M. Braun, H. Friepertinger, A. Kerber, A. Kohnert, and A. Wassermann, *Error-Correcting Linear Codes: Classification by Isometry and Applications*, ser. Algorithms and Computation in Mathematics. Springer Berlin Heidelberg, 2006.
- [35] B. Schmalz, “ t -Designs zu vorgegebener Automorphismengruppe,” *Bayreuther Mathematische Schriften*, no. 41, pp. 1–164, 1992, Dissertation, Universität Bayreuth, Bayreuth.
- [36] *GAP – Groups, Algorithms, and Programming, Version 4.7.7*, The GAP Group, 2015. [Online]. Available: <http://www.gap-system.org>
- [37] T. Rehn and A. Schürmann, “C++ tools for exploiting polyhedral symmetries,” *Lecture Notes in Computer Science*, vol. 6327/2010, 2010.
- [38] John MacLaren Walsh and Congduan Li, “Network Enumeration and Hierarchy in GAP,” [Online]. Available: <http://www.ece.drexel.edu/walsh/asptgrg/software.html>
- [39] C. Li, J. M. Walsh, and S. Weber, “TransIT 2015 Data: Enumeration and Rate Regions for General Hyperedge Networks,” available at <https://goo.gl/yZfD9>.
- [40] R. W. Yeung and Y. O. Yan, “Information theoretic inequality prover,” 2008, <http://user-www.ie.cuhk.edu.hk/ITIP/>.
- [41] E. Rethnakaran Pulikoonattu and Suhas Diggavi, “X-information theoretic inequality prover,” 2008, <http://user-www.ie.cuhk.edu.hk/ITIP/>.
- [42] L. Csirmaz, “a MINimal Information Theoretic Inequality Prover,” <https://github.com/lcsirmaz/minitip>.
- [43] Zhen Zhang and Raymond W. Yeung, “A Non-Shannon-Type Conditional Inequality of Information Quantities,” *IEEE Trans. on Information Theory*, vol. 43, no. 6, Nov. 1997.
- [44] R. Dougherty, C. Freiling, and K. Zeger, “Networks, Matroids, and Non-Shannon Information Inequalities,” *IEEE Trans. on Information Theory*, vol. 53, no. 6, pp. 1949–1969, Jun. 2007.
- [45] H. Sun and S. A. Jafar, “Index coding capacity: How far can one go with only shannon inequalities?” *IEEE Transactions on Information Theory*, vol. 61, no. 6, pp. 3041–3055, June 2015.
- [46] A. S. N. V. D. Hammer, A. Romashchenko, “Inequalities for Shannon Entropy and Kolmogorov Complexity,” *Journal of Computer and System Sciences*, no. 60, pp. 442–464, 2000.
- [47] Randall Dougherty, Chris Freiling, Kenneth Zeger, “Linear rank inequalities on five or more variables,” submitted to SIAM J. Discrete Math. arXiv:0910.0284.
- [48] Randall Dougherty, Chris Freiling, and Ken Zeger, “Insufficiency of linear coding in network information flow,” *IEEE Trans. Inf. Theory*, vol. 51, no. 8, pp. 2745–2759, Jul. 2005.
- [49] A. W. Ingleton, “Representation of Matroids,” in *Combinatorial Mathematics and its Applications*, D. J. A. Welsh, Ed. San Diego: Academic Press, 1971, pp. 149–167.
- [50] T. Ho, M. Effros, and S. Jalali, “On equivalence between network topologies,” in *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Sept 2010, pp. 391–398.
- [51] X. Xu, S. Thakor, and Y. L. Guan, “Reduced functional dependence graphs and their applications,” in *2012 International Symposium on Network Coding (NetCod)*, June 2012, pp. 61–66.
- [52] N. Robertson and P. Seymour, “Graph minors. i. excluding a forest,” *Journal of Combinatorial Theory, Series B*, vol. 35, no. 1, pp. 39 – 61, 1983.
- [53] —, “Graph minors. xx. wagner’s conjecture,” *Journal of Combinatorial Theory, Series B*, vol. 92, no. 2, pp. 325 – 357, 2004, special Issue Dedicated to Professor W.T. Tutte.
- [54] P. D. Seymour, “Matroid Representation over $\text{GF}(3)$,” *J. Combin. Theory Ser. B*, no. 26, pp. 159–173, 1979.
- [55] James Oxley, *Matroid Theory, 2nd. Ed.* Oxford University Press, 2011.
- [56] J. Geelen, B. Gerards, and G. Whittle, “Solving rota’s conjecture,” *Notices of the American Mathematical Society*, pp. 736–743, 2014.
- [57] Congduan Li, Steven Weber, and John MacLaren Walsh, “On Multilevel Diversity Coding Systems,” *IEEE Trans. Inf. Theory*, vol. 63, no. 1, pp. 230–251, Jan. 2017. [Online]. Available: <https://doi.org/10.1109/TIT.2016.2628791>
- [58] C. Li, J. M. Walsh, and S. Weber, “Software for computing bounds on entropic vectors region and network rate region,” available at <http://www.ece.drexel.edu/walsh/asptgrg/software.html>.
- [59] —, “ITW 2015 Data: Enumeration and Exact Rate Regions for Networks,” available at <http://goo.gl/f1spQz>.
- [60] *ITAP – Information Theoretic Achievability Prover*, Adaptive Signal Processing and Information Theory Research Group, 2015. [Online]. Available: <http://www.ece.drexel.edu/walsh/asptgrg/software.html>
- [61] *ITCP – Information Theoretic Converse Prover*, Adaptive Signal Processing and Information Theory Research Group, 2016. [Online]. Available: <http://www.ece.drexel.edu/walsh/asptgrg/software.html>

APPENDIX A PROOF OF THEOREM 2

In the interest of conciseness, for all but **(D4)** and **(D8)** we will only briefly sketch the proof for the expressions determining $\mathcal{R}_*(A')$ from $\mathcal{R}_*(A)$, as the map in the opposite direction and the other rate region bounds follow directly from parallel arguments.

(D1) holds because s' is not communicating with any nodes other than possibly sinks. If there is a sink that demands it that does not have direct access to it, then this sink can not successfully receive any information from it, since s' does not communicate with any intermediate nodes. Hence, in this case $\omega_{s'} = 0$ and every other rate is constrained according to $\mathcal{R}_*(A)$ because the remainder of the network has no interaction with s' . Alternatively, if every sink that demands s' has direct access to it, any non-negative source rate can be supported for s' , and the remainder of the network is constrained as by $\mathcal{R}_*(A)$ because no other part of the network interacts with s' . **(D2)** holds because the demand of s' at sink t' is trivially satisfied if it has direct access to s' . The constraint has no impact on the rate region of the network.

In **(D3)** if a source is not demanded by anyone, it can trivially support any rate.

When two sources have exactly the same connections and are demanded by same sinks as under **(D4)**, they can be simply viewed as a combined source for \mathcal{R}_l with $l \in \{c, *, q, o\}$, since the exact region and these bounds enable simple concatenation of sources. Since the source entropies are variables in the rate region expression, it is equivalent to make s as the combined source, which since the previous sources were independent, will have an entropy which is the sum of their entropies. Moving from $\mathcal{R}_l(A')$ to $\mathcal{R}_l(A)$ is then accomplished for any $l \in \{c, *, q, (s, q), o\}$ by observing that A can be viewed as A' with $\omega_{s'} = 0$.

An intermediate node can only utilize its input hyperedges to produce its output hyperedges, hence when two intermediate nodes have the same input edges, their encoding capabilities are identical, and thus for pursuing minimality of representation of a network, these two nodes having the same input should be represented as one node. Thus, **(D5)** is necessary and the merge of nodes with same input does not impact the coding on edges or the rate region, as the associated constraints $\mathcal{L}_A = \mathcal{L}_{A'}$.

If the input or output of an intermediate node is empty, as in **(D6)** it is incapable of affecting the capacity region. If, as in the second case covered by **(D6)** the input to an sink node is empty, any sources which it demands can only be reliably decoded if they have zero entropy.

(D7) is clear because an edge to nowhere can not effect the rest of the capacity region and is effectively unconstrained itself.

(D8) can be shown as follows. If $\mathcal{R}_*(A)$ is known and when edge e in A is represented as two parallel edges e, e' so that the network becomes A' , then the constraint on e, e' in A' is simply to make sure the total capacity $R_e + R_{e'}$ can allow the information to be transmitted from the tail node to head nodes. Simple concatenation of the messages among the two edges will achieve this for those bounds $l \in \{c, *, q, o\}$ allowing such concatenation. Therefore, replace the R_e in $\mathcal{R}_l(A)$ with $R_e + R_{e'}$ will obtain the rate region $\mathcal{R}_l(A')$ for any $l \in \{c, *, q, o\}$. Moving from $\mathcal{R}_l(A')$ to $\mathcal{R}_l(A)$ is accomplished by recognizing that A is effectively A' with $R_{e'} = 0$.

Under the condition in (D9), an intermediate node g' has exactly one input edge e and exactly one output hyperedge e' , and the input e is an edge (i.e. g' is its only destination). The rate coming out of this node can be no larger than the rate coming in since the single output hyperedge must be a deterministic function of the input edge. It suffices to treat these two edges as one hyperedge connecting the tail of e to the head of e' with the rate the minimum of the two rates.

If a sink demands a source that it does not have access to, the only way to satisfy this network constraint is the source entropy is 0. Hence, (D10) holds. The removal of this redundant source does not impact the rate region of the network with remaining variables.

(D11), similar to (D3), observes that two sink nodes with same input yield the same constraints $\mathcal{L}_{A'}$ as \mathcal{L}_A with them merged.

(D12) is easy to understand because the decoding ability of $\beta(t)$ at sink node t is implied by sink t' . The non-necessary repeated decoding constraints will not affect the rate region.

(D13), as (D12), observes that the ability of t to decode s' implies that t' can decode it as well. Adding or removing the direct access to s' at t' will not affect the rate region.

(D14) is obviously true since the weakly disconnected components can not influence each others rate regions.

APPENDIX B PROOF OF THEOREM 4

We will prove $\mathcal{R}_l(A') = \text{Proj}_{\omega, r \setminus R_e}(\{\mathbf{R} \in \mathcal{R}_l(A)\})$ for $l \in \{*, q, o\}$, and for the scalar case, $\mathcal{R}_{s,q}(A'') \supseteq \text{minimal}_{A' \rightarrow A''}(\text{Proj}_{\omega, r \setminus R_e} \mathcal{R}_{s,q}(A))$, since the remainder of the theorem holds from the minimality reductions in Thm. 2.

Select any point $\mathbf{R}' \in \mathcal{R}_*(A')$. Then there exists a conic combination of some points in $\mathcal{R}_*(A')$ that are associated with entropic vectors in $\Gamma_{N'}^*$ such that $\mathbf{R}' = \sum_j \alpha_j \mathbf{r}'_j$, where $\alpha_j \geq 0, \forall j$. For each \mathbf{r}'_j , there exist random variables $\mathbf{Y}_S^{(j)}, U_i^{(j)}, i \in \mathcal{E} \setminus e$, such that the entropy vector

$$\mathbf{h}^{(j)'} = \left[H(\mathcal{A}) \mid \mathcal{A} \subseteq \left\{ Y_s^{(j)}, U_i^{(j)} \mid s \in \mathcal{S}, i \in \mathcal{E} \setminus e \right\} \right]$$

is in $\Gamma_{N'}^*$, where $N' = N - 1$ is the number of variables in A' . Furthermore, their entropies satisfy all the constraints determined by A' . In the network A , define $U_e^{(j)}$ to be the concatenation of all inputs to the tail node of e , $U_e^{(j)} = \mathbf{U}_{\text{In}(\text{TI}(e))}^{(j)}$. Then the entropies of random variables $\{\mathbf{Y}_S^{(j)}, U_{\mathcal{E}}^{(j)}\}$ will satisfy the constraints in A , and additionally obey $H(U_e^{(j)}) = H(\mathbf{U}_{\text{In}(\text{TI}(e))}^{(j)})$. Hence, $\mathbf{h}^{(j)} = \left[H(\mathcal{A}) \mid \mathcal{A} \subseteq \left\{ Y_s^{(j)}, U_i^{(j)} \mid s \in \mathcal{S}, i \in \mathcal{E} \right\} \right] \in \Gamma_N^*$. That is,

$\mathbf{r}_j = [\mathbf{r}'_j, R_e \geq H(\mathbf{U}_{\text{In}(\text{TI}(e))}^{(j)})] \in \mathcal{R}_*(A)$. By using the same conic combination, we have an associated rate point $\mathbf{R} = \sum_j \alpha_j \mathbf{r}_j \in \left\{ \mathbf{R} \in \mathcal{R}_*(A) \mid R_e \geq H(\mathbf{U}_{\text{In}(\text{TI}(e))}^{(j)}) \right\}$. Thus, $\mathcal{R}_q(A') \subseteq \text{Proj}_{\omega, r \setminus R_e}(\{\mathbf{R} \in \mathcal{R}_*(A) \mid R_e \geq H(\mathbf{U}_{\text{In}(\text{TI}(e))}^{(j)})\})$, which in turn, is $\subseteq \text{Proj}_{\omega, r \setminus R_e} \mathcal{R}_*(A)$.

If \mathbf{R}' is achievable by general \mathbb{F}_q codes, since concatenation of all input is a valid \mathbb{F}_q vector code, we have $\mathcal{R}_q(A') \subseteq \text{Proj}_{\omega, r \setminus R_e}(\{\mathbf{R} \in \mathcal{R}_q(A) \mid R_e \geq H(\mathbf{U}_{\text{In}(\text{TI}(e))}^{(j)})\})$ which in turn is $\subseteq \text{Proj}_{\omega, r \setminus R_e} \mathcal{R}_q(A)$.

However, we cannot establish same relationship when scalar \mathbb{F}_q codes are considered, because for the point \mathbf{R}' , the associated \mathbf{R} with $H(U_e)$ may not be scalar \mathbb{F}_q achievable.

On the other hand, if we select any point $\mathbf{R} \in \{\mathbf{R} \in \mathcal{R}_*(A)\}$, then, there exists a conic combination of some points in $\mathcal{R}_*(A)$ associated with entropic vectors in Γ_N^* , i.e., $\mathbf{R} = \sum_j \alpha_j \mathbf{r}_j$, $\alpha_j \geq 0, \forall j$. For each \mathbf{r}_j , there exist random variables $\{\mathbf{Y}_S^{(j)}, U_{\mathcal{E}}^{(j)}\}$ such that their entropies satisfy all the constraints determined by A . Since the entropies of $\{\mathbf{Y}_S^{(j)}, U_i^{(j)} \mid i \in \mathcal{E} \setminus e\}$ satisfy all constraints determined by A' (because they are a subset of the constraints from A) and the entropic vector projecting out U_e is still entropic. Thus, by letting $R_e^{(j)}$ to be unconstrained, we have $\text{Proj}_{\omega, r \setminus R_e} \mathbf{r}_j \in \mathcal{R}_*(A')$. Further, by using the same conic combination, $\mathbf{R}' = \text{Proj}_{\omega, r \setminus R_e} \sum_j \alpha_j \mathbf{r}_j = \text{Proj}_{\omega, r \setminus R_e} \mathbf{R} \in \mathcal{R}_*(A')$. Thus, we have $\text{Proj}_{\omega, r \setminus R_e}(\{\mathbf{R} \in \mathcal{R}_*(A)\}) \subseteq \mathcal{R}_*(A')$.

If $\mathbf{R} \in \mathcal{R}_*(A)$ is achievable by \mathbb{F}_q code \mathbb{C} , either scalar or vector, then the code to achieve $\mathbf{R}' = \text{Proj}_{\omega, r \setminus R_e} \mathbf{R} \in \mathcal{R}_*(A')$ could be the code \mathbb{C} with deletion of columns associated with edge e , i.e., $\mathbb{C}' = \mathbb{C}_{:, \setminus U_e}$, because the code on edge e is not of interest. Thus, we have $\text{Proj}_{\omega, r \setminus R_e} \mathcal{R}_l(A) \subseteq \mathcal{R}_l(A')$, $l \in \{q, (s, q)\}$.

Furthermore, for any point $\mathbf{R}' \in \mathcal{R}_o(A')$, there exists an associated point $\mathbf{h}' \in \Gamma_{N'}$ and a rate vector $\mathbf{r}' = [R_i \mid i \in \mathcal{E} \setminus e]$ such that $\mathbf{R}' = \text{Proj}_{\omega, r \setminus R_e} [\mathbf{h}', \mathbf{r}'] \cap \mathcal{L}_{A'}$. Clearly, if we increase the dimension of \mathbf{h}' by adding a variable U_e which is the vector of all input variables to the tail node of e , i.e., $U_e = [U_i \mid i \in \text{In}(\text{TI}(e))]$ and $H(U_e) = H(U_i, i \in \text{In}(\text{TI}(e)))$, we have the new vector in Γ_N . That is, if we define

$$\mathbf{h} = \begin{cases} h'_{A \cap \{Y_s, U_i \mid s \in \mathcal{S}, i \in \mathcal{E}'\}}, & U_e \notin \mathcal{A} \\ h'_{A \cap \{Y_s, U_i \mid s \in \mathcal{S}, i \in \mathcal{E}'\} \cup \{U_i \mid i \in \text{In}(\text{TI}(e))\}}, & U_e \in \mathcal{A} \end{cases} \quad (30)$$

for $\mathcal{A} \subseteq \{Y_s, U_i \mid s \in \mathcal{S}, i \in \mathcal{E}\}$, then $\mathbf{h} \in \Gamma_N$. Further, we let R_e to be unconstrained, i.e., $R_e = \infty$. Since $H(U_e) \leq R_e$, the network constraints in A will be satisfied given that the other constraints will not be affected. Hence, there exists an associated point $\mathbf{R} \in \mathcal{R}_o(A)$ with $H(U_e) \leq R_e$, where R_e is unconstrained. Therefore, we have $\mathcal{R}_o(A') \subseteq \text{Proj}_{\omega, r \setminus R_e}(\{\mathbf{R} \in \mathcal{R}_o(A)\})$. Reversely, suppose a point $\mathbf{R} \in \mathcal{R}_o(A)$ is picked with R_e unconstrained. There exists an associated vector $\mathbf{h} \in \Gamma_N$ and a rate vector $\mathbf{r} = [R_i \mid i \in \mathcal{E}]$ such that $\mathbf{R} = \text{Proj}_{\omega, r} [\mathbf{h}, \mathbf{r}] \cap \mathcal{L}_A$. Since R_e is unconstrained, we will have $H(U_e)$ unconstrained as well. Since the network constraints \mathcal{L}_A with R_e unconstrained will be $\mathcal{L}_{A'}$, and $\text{Proj}_{\omega, r \setminus R_e} [\mathbf{h}, \mathbf{r}] \in \Gamma_{N'} \cap \mathcal{L}_{A'}$, we have $\text{Proj}_{\omega, r \setminus R_e} \mathbf{R} \in \mathcal{R}_o(A')$. Therefore, we have $\text{Proj}_{\omega, r \setminus R_e}(\{\mathbf{R} \in \mathcal{R}_o(A)\}) \subseteq \mathcal{R}_o(A')$.